Microsoft

Programování v prostředí MakeCode

Pololetní výukový kurz

O tomto kurzu

Informace o kurzu

Tento kurz obsahuje výuku na jedno pololetí. Je určený pro žáky ve věku 11-15 let a má sloužit jako úvod do IT, především do programování.

Tento materiál je vhodný pro učitele, kteří dosud nikdy neučili IT.

V průběhu tohoto kurzu se budou žáci seznamovat s koncepty programování s využitím programovacího prostředí Microsoft MakeCode a hry Minecraft: Education Edition.

Všechny zdrojové kódy naleznete také na minecraftedu.cz.

Obsah

O lekcích

Každá lekce tohoto kurzu začíná vysvětlením tématu.

V každé kapitole naleznete následují tři odlišné typy aktivit:

1. "Offline" aktivity – bez použití zařízení, počítače
2. Řízené aktivity – připravené krok za krokem
3. Samostatný projekt

Tyto aktivity se navzájem doplňují a podporují. I když každý typ pracuje s jiným konceptem, vytváří jiné výzvy a rozvíjí tak kreativitu a další dovednosti žáků:

	Offline aktivity	**Řízené aktivity**	**Projekt**
Pojmy	Intenzivní	Průměrné	Málo
Rozvoj dovedností	Málo	Intenzivní	Průměrné
Kreativita	Průměrné	Málo	Intenzivní

Offline aktivity

Offline aktivity jsou dobrou cestou, jak uvést nové pojmy zábavným způsobem, který zvedne žáky od počítačů a přivede k vzájemné interakci s ostatními ve skupině. Je nezbytné představit pojem tak, aby měli žáci představu o problému dříve, než budou objevovat principy skrze Microsoft MakeCode a Minecraft.

Řízené aktivity

Jde o aktivity, které postupují krok po kroku podle návodných instrukcí, kde všichni žáci provádějí ty samé věci. Tak, jako když se v technické výchově postupně vyrábí ptačí budka, postupují žáci od začátku pomocí jednoduchých kroků až k vytvoření finálního výrobku.

Žáci mohou pracovat každý samostatně. Pokud budou postupovat rychleji než ostatní, mohou po dokončení zadání realizovat další svá vylepšení. Tyto aktivity rozvíjejí intenzivně dovednosti, ale dávají málo prostoru tvořivosti, protože všichni dělají totéž. Proto je důležité neskončit pouze tímto typem aktivity a dát prostor dalšímu typu.

Samostatné projekty

Každá lekce vrcholí vytvořením samostatného projektu, který dává prostor žákovi ukázat, co už umí. V této fázi se už zpravidla nepotřebují učit nové dovednosti. "Pouze" demonstrují, jak zvládli téma pomocí řízených aktivit v kontextu jiného projektu. Některé projekty jsou navržené tak, aby byly pro více žáků a oni tak museli spolupracovat a tvořit v týmech. Kreativní projekty umožňují žákům využít nové dovednosti a naučené postupy jedinečným způsobem. Umožněte jim navzájem sdílet jejich nápady a řešení!

V průběhu samostatného projektu by si žáci měli psát deník (ideálně elektronicky – např. ve Wordu nebo OneNote). Na konci každého projektu jsou vypsané návodné otázky, na které by žáci měli ve svém deníku odpovědět. Pod otázkami je pro učitele připravené i možné bodové hodnocení.

O Autorech

Douglas King

Douglas Kiang je přednášející a učitel s více jak osmadvacetiletou praxí. Působil na různých školách všech stupňů vzdělávání. V současné době učí středoškolskou informatiku na Punahou School na Honolulu na Hawai. Douglas získal magisterský titul v oblasti technologií, inovace a vzdělávání na harvardské univerzitě. Je členem skupiny Microsoft Innovative Educator a Minecraft Global Mentor. Můžete ho sledovat na Twitteru pod @dkiang.

Mary King

Mary Kiang vyučuje více jak 25 let na základní, střední a vysoké škole. Rovněž vypracovala učební osnovy ve Vzdělávacím oddělení vědeckého muzea v Bostonu. V současné době učí v šesté třídě matematiku a přírodní vědy na Punahou School. Mary je bývalou programátorkou pro Houghton Mifflin a Dun & Bradstreet a je držitelem magisterského titulu pro základní vzdělávání ze Simmons College. Mary je Minecraft Global Mentor a navíc je také zakladatelkou GO Code! - organizace, která podporuje dívky a mladé ženy při jejich začátcích v programování a STEM. Můžete ji sledovat na Twitteru pod @maryrkiang.

Přeložili a upravili

Miroslav Kotlas

Miroslav Kotlas je učitelem informatiky na Gymnáziu Česká v Českých Budějovicích. Léta aktivně zapojuje moderní technologie a inovativní přístupy do své výuky. Specializuje se na zapojení počítačové hry Minecraft do výuky. Ve svých hodinách ho využívá převážně k výuce algoritmizace. Pro české učitele vytváří a překládá materiály pro výuku v Minecraft: Education Edition. Celou řadu jeho zdrojových kódů a přeložených světů můžete nalézt na stránce minecraftedu.cz. Je také členem skupiny Microsoft Innovative Educator.

Tento materiál by v češtině nevznikl bez pomoci studentů Gymnázia Česká v Č. Budějovicích, především pak Marka Maňáska a Radka Bláhy, kterým patří velký dík za testování kódu, vytvoření obrázků kódů v českém jazyce i za kontrolu chyb.

Seznámení s Minecraftem a prostředím Makecode

O Minecraftu

Co je Minecraft? Jednoduše řečeno, Minecraft je největší sandboxová hra, kterou jste kdy hráli. Její krajina se nekonečně roztahuje do všech směrů, jak daleko jen oko dohlédne. Sahá nahoru do hor a kopců a dolů do jeskyň plných lávy a pokladů. Stejně jako v sandboxu, Minecraft nemá žádný daný cíl, žádné předurčené úkoly k naplnění. Zábava spočívá v určení svých vlastních cílů a rozhodování, jak chcete hrát. Možná budete chtít vyrazit do džungle hledat lesní chrám. Třeba si postavíte svůj vysněný dům na stromě s perfektním výhledem na oceán, nebo se můžete prokopat hluboko pod zem a hledat smaragdy a diamanty. Kdykoliv se jeden úkol stane moc těžký nebo nudný, nebo se někde zaseknete a budete potřebovat pomoc nebo inspiraci, můžete dělat něco jiného. Vždy se najde něco poutavého. Nezáleží na tom, co budete chtít dělat, vždy se naučíte něco o světě kolem vás a budete přemýšlet o lepších a účinnějších cestách, jak váš cíl splnit.

Na první pohled není Minecraft vizuálně působivý. Noví hráči se často zaseknou nad tím, jak „hranatě“ všechno vypadá. Ale po chvíli hraní se do herního prostředí ponoří a uvědomí si komplexnost herního světa. Jsou to právě situace a výzvy, které se zdají až reálné – ne grafika či animace. Ve skutečnosti tato relativně surová grafika dodává Minecraftu kreslený pocit, který je svým vlastním způsobem okouzlující a odzbrojující a celkově odlehčuje pocit ze hry.

Učení Minecraftem je jako učení bloky. Právě bloky jsou poutavý prostředek pro hodiny, které kolem nich postavíte. Můžete nechat vaše žáky si s bloky volně hrát a podívat se na to, co vymyslí a sdílet jejich výtvory. Můžete bloky použít k vystavění imaginárního městečka a vést rozhovor o tom, kam mají vést ulice, jak by se které budovy měly jmenovat a jestli budete nebo nebudete potřebovat vězení.

Programování v Minecraftu je jako obdržení superschopností. Žáci jsou již zvyklí na to, jak Minecraft běžně funguje. Představa použití programu k automatizování zdlouhavých úkolů jako těžení nebo kácení stromů je vůči hráčům velice motivující a možnost okamžitě vidět výsledek kódu, který jste v Minecraftu naprogramovali, je neuvěřitelně silná. Žáci mohou na začátek použít pro jejich program plochý svět a v průběhu učení a experimentování budou moci vizuálně vidět postup v jejich světě jako sérii různých fyzických projektů a staveb.

Programovat můžete jak v Minecraft: Education Edition, tak i v Minecraft for Windows 10.

Na internetu najdete spoustu návodů a videí o tom, jak Minecraft hrát. Tady je pár z nich:

- http://minecraftedu.cz
- https://www.kidscodr.cz/
- https://minecraft-cs.gamepedia.com/Návody

O MakeCode

V roce 1975 Seymour Papert z MIT Media Lab vytvořil programovací jazyk pro začátečníky – LOGO. Vypracoval ho na základě výzkumu, který ukazoval, že hraní si s bloky kódu je velice efektivní způsob učení programovacích konceptů. Papert vytvořil termín „konstrukcionismus" k popsání způsobu, kterým si mohou žáci vytvářet nové poznatky budováním na těch starých. Bloky v MakeCode a bloky v Minecraftu jsou samy o sobě modelem pro způsob učení, kde se získávání zkušeností děje skrz aplikaci konceptů v otevřeném učebním prostředí. Programovací jazyky, které jsou založené na blocích jako MakeCode a vystavěné na Papertově výzkumu, jsou pro žáky skvělou cestou, jak se učit o nových způsobech programování, navíc bez nutnosti starat se o syntax.

Stejně jako je možné použít vizuální Drag-and-Drop blokové prostředí MakeCode, tak je podporován i textový editor JavaScript nebo Python. Jak se žáci stávají sebejistější, mohou postoupit z bloků do programování „z reálného světa". Tento kurz je založen hlavně na programování z bloků, ale jsou v něm obsaženy některé lekce (Aktivita: Vytvoření bludiště), kde se žáci podívají na kód v JavaScriptu. Pokud máte žáky, kteří jsou v programování již pokročilejší, mohou plnit úkoly v JavaScriptu hned od začátku.

Pokud chcete o MakeCode a dalších produktech, které podporuje, zjistit více, navštivte MakeCode.com

Události

O událostech

V této lekci se seznámíme s událostmi a jejich zachytávači, které tvoří důležitý koncept v IT a které lze nalézt ve všech programovacích jazycích. Začneme aktivitou bez počítače, kde si vysvětlíme a ukážeme systém příčiny a reakce na nastalou situaci tak, jak to normálně funguje v běžném životě. Dále budeme pracovat s MakeCodem v Minecraftu. Nakonec si každý vytvoří vlastní projekt v Makecode, který bude využívat události k aktivaci připravených částí programu.

Události

"Událost" je v IT akce nebo situace, která je rozpoznána počítačem nebo programem. Když někdo klikne tlačítkem myši, vznikne (vygeneruje se) v operačním systému počítače událost „mouse click event". V normálním světě známe tento mechanismus pod spojeními příčina-následek nebo akce-reakce.

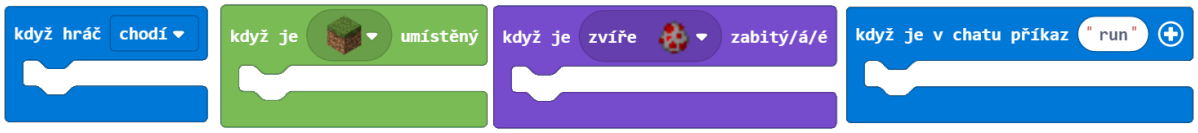
Zde je několik příkladů:

Událost		Akce
Začne pršet	➔	Otevřu si deštník
Zazvoní zvonek	➔	Žáci jdou do tříd
Je stisknuto tlačítko napájení	➔	Počítač se zapne

Je stisknuto tlačítko myši	➔	Otevře se aplikace

Můžete doplnit ještě nějaké další události a přidat k nim odpovídající reakce na ně?

V programování je zachytávač události (event handler) částí vašeho programu, který spustí vámi napsanou část programu, jež jste naplánovali, aby se stala v situaci, kdy nastane právě tato událost (tzn. odchytí událost; „handles“ the event). V MakeCode tyto bloky zachytávačů událostí vypadají jako rámečky s mezerou uprostřed, zobáčkem pro přichycení dalších bloků příkazů a obvykle text v záhlaví začíná slovem „když“:



Wikipedia Event-driven programming

Offline aktivita: Události a jejich zachytávače

Poznámka pro učitele:

Offline aktivita je aktivita, která nevyužívá počítačů ani jiné techniky – je tzv. odpojena od zařízení. Obvykle se snažíme představit nové koncepty zábavnou formou, která nutí žáky k pohybu, často reagovat a spolupracovat se spolužáky tváří v tvář, zatímco hrají hry nebo řeší úkoly. Offline aktivity dovolují žákům vyzkoušet si koncepty a pochopit pojmy bez počítačů. Když poté usednou zpět k počítačům, již nad danou problematikou přemýšleli a mají s ní vlastní zkušenosti.

Cíl:

Znázornit model událostmi řízeného programování vytvářením běžných událostí a jejich řešením pomocí reakcí zachytávačů událostí.

Přehled:

Rozdělte žáky tak, aby jedna třetina představovala tým „Události“ a zbytek bude tvořit tým „Zachytávači událostí“.

Tým „Události“ by měl sám vytvářet události. Žáci by měli vymýšlet své vlastní události, jako příklad jim můžeme uvést:

- Dveře se otevřely.
- Světla se zhasla.
- Tleskli jsme dvakrát rukama.
- Zvedli jsme obě ruce.

Takto vymyšlenou událost žáci napíší na 2 kartičky. Tým připraví několik událostí tak, aby každý mohl představit alespoň jednu událost a aby jejich počet byl dostatečný, ale aby jejich představení nezabralo příliš mnoho času. Kartičky se zamíchají a předají se týmu „Zachytávači událostí.“

Když „Zachytávači událostí“ dostanou připravené kartičky s událostmi, rozdělí si je mezi sebe. Žáci by pak měli přijít s návrhem své vlastní odpovídající reakce. Tady jsou uvedené příklady pro vysvětlení:

- Dveře se otevřely -> Vyjdeme ven
- Světla se zhasla -> Půjdeme spát
- Tleskli jsme dvakrát rukama -> Třikrát dupneme
- Zvedli jsme obě ruce -> Řekneme: „Gól!“

Postavte tým „Události“ dopředu v místnosti tak, aby na ně bylo dobře vidět. Každý z týmu pak postupně zrealizuje, tj. provede nebo předvede, svou událost. Po „vykonání“ události by měli žáci z týmu „Zachytávačů událostí“, kteří měli tuto událost zpracovat, představit nebo provést svou reakci na tuto událost.

Až se takto použijí všechny připravené události, můžete vymyslet nové události a žáci na ně mohou vymýšlet reakce.

Materiál:

- Kartičky & tužky

Pravidla:

- Pokud není řečeno vámi jinak, nebudou žáci, kteří zrovna nic nepředvádějí, rušit.
- Žáci by měli dávat pozor na aktivitu předvádějících, ale neměli by ji navzájem komentovat nebo si radit.

Reflexe:

Promluvte si o tom, jak se žáci cítili a jak se jim pracovalo:

- Vyskytly se nějaké programátorské chyby v systému? Zapomněl některý žák reagovat na událost?
- Jaké to bylo sledovat ostatní předvádějící?
- Občas se stalo, že na jednu událost reagovalo více lidí naráz. Jak to fungovalo?
- Může fungovat nebo existovat jeden zachytávač událostí pro více událostí? (Ano)
- Může nějaká reakce na zachycení události vyvolat spuštění jiného zachytávače událostí? (Ano) A jestli ano, jak by to mělo fungovat? (Světla se vypnula -> Jdu spát -> Učitel říká „Vzbuď se!“)

Tipy:

- BEZPEČNOST PŘEDEVŠÍM! Žáci, zvláště ti mladší, mohou občas jednat bláznivě. A přestože to má být především zábava, je třeba pamatovat na to, že na prvním místě je bezpečnost!

Poznámky:

Propojení s programováním počítačů: Počítačové programy jsou sady instrukcí, které říkají počítači, jak zpracovat vstup a poskytnout výstup. Důležitou součástí programování je říct počítači, KDY má vykonat nějakou činnost. Vytváření událostí a jejich zachytávání je způsobem, jak spouštět určité části programu.

Aktivita: Chodník ze zlatých bloků

Poznámka pro učitele:

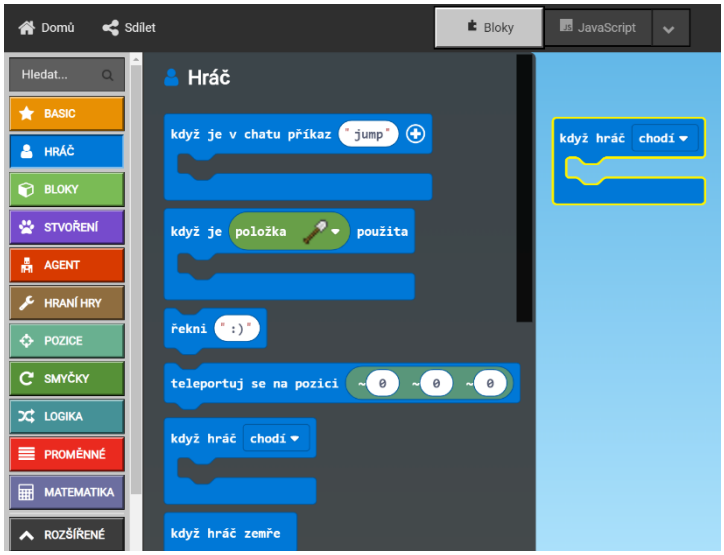
Řízené aktivity vzaly svou předlohu z hodin pracovní výchovy. Tak, jako když se v technické výchově postupně vyrábí ptačí budka, postupují žáci od začátku pomocí jednoduchých kroků až k vytvoření finálního výrobku. Pokud všichni postaví alespoň trochu slušně vypadající budku, víte, že mají za sebou alespoň základy práce se dřevem a naučili se něco nového. Jakmile víte, že žáci získali základní dovednosti, můžete se přesunout ke kreativnějším a otevřenějším projektům. Nekončete řízenými aktivitami! Ačkoli je nejjednodušší hodnotit aktivity tohoto typu, ujistěte se, že žáci mají příležitost aplikovat nové dovednosti při splnění otevřenějších a kreativnějších úkolů.

Blok „Když hráč chodí“ je zachytávač události, který čeká na konkrétní akci hráče. V horní části tohoto bloku se nachází rozbalovací seznam celé řady akcí hráče, které může kdykoliv během hry vykonat. Například chodit, lézt, plavat v lávě atd.

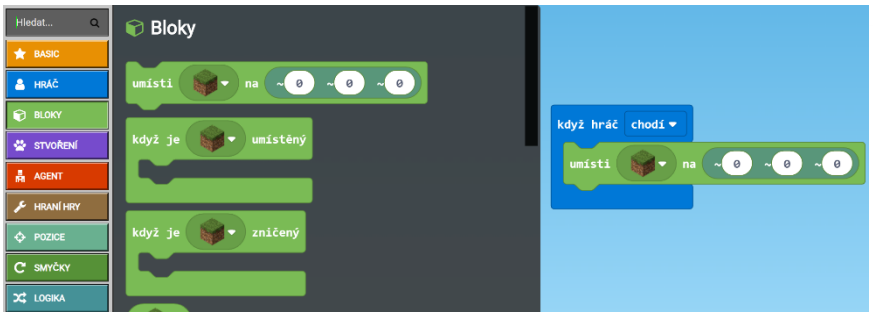
Můžete nakonfigurovat zachytávač této události tak, aby něco vykonal, když hráč chodí. Například může během své chůze rozhazovat květiny!

Kroky:

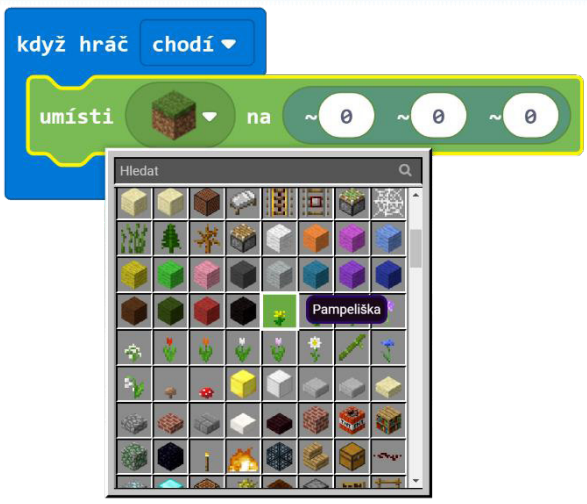
1. Z nabídky Hráč, přetáhněte blok Když hráč chodí do pracovního prostoru.



2. Z nabídky Bloky, přetáhněte blok Umísti na dovnitř do bloku Když hráč chodí, až uslyšíte a uvidíte jeho připojení na správné místo.



3. Použijte rozbalovací seznam v bloku Umísti na a vyberte žlutou kytku (pampelišku).



Od teď, kdykoliv budete ve hře chodit, budou z vás padat pampelišky! Zkuste to ve hře tak, že budete používat ovládací klávesy pro chození W S A a D. Pak se podívejte za sebe – měli byste vidět cestičku z kytiček!

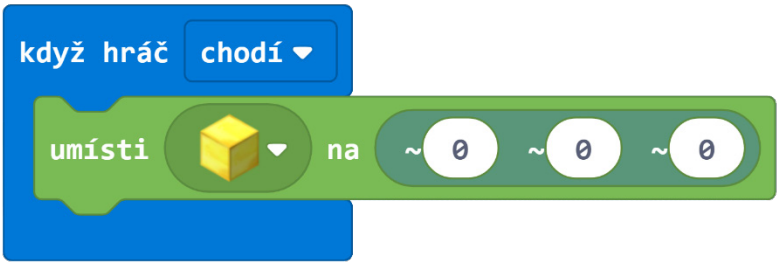
Poznámka pro učitele:

V příští lekci budeme podrobněji zkoumat souřadný systém hry.

Pro teď nám bude stačit vědět, že se používají 3 souřadnice (X, Y, Z), které představují různé směry světa Minecraftu:

- Souřadnice X – Východ/Západ
- Souřadnice Y – Nahoru/Dolů
- Souřadnice Z – Sever/Jih

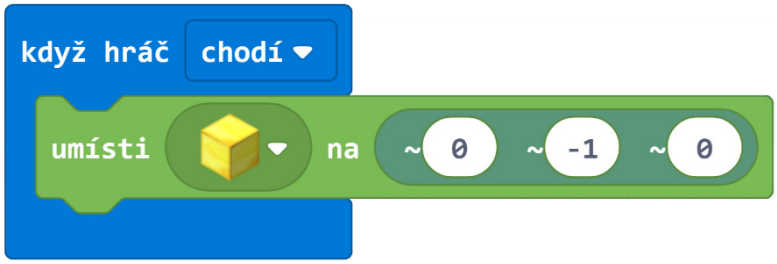
Je nutné upozornit, že jsme k umístění našeho bloku použili souřadnice (~0 ~0 ~0). ~0 na druhé (prostřední) pozici představuje relativní pozici k úrovni země (pozice, kde stojíme). Předpokládejme, že chceme za sebou vytvořit zlatou cestu tak, že budeme jako hráč chodit krajinou. Můžeme k tomu použít ten samý blok umísti na. Pouze místo kytiček budeme umisťovat pevné zlaté bloky.



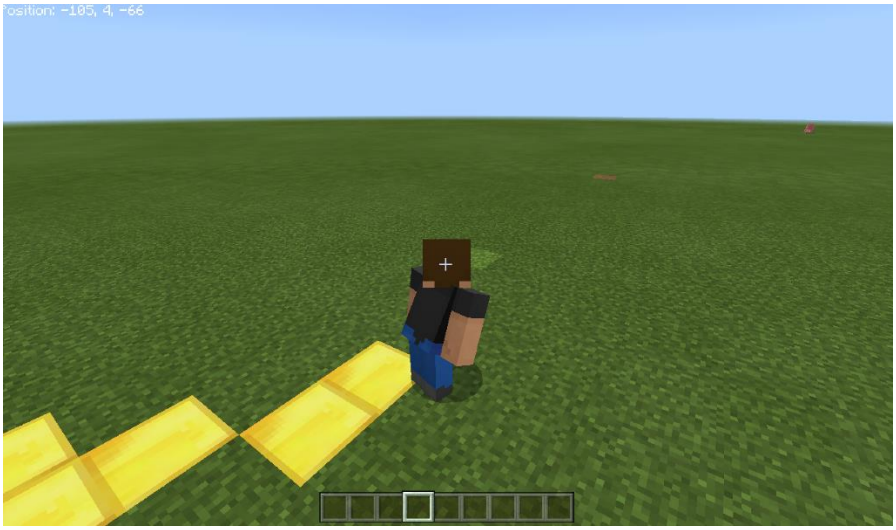
Sledujte, co se stane:

Dobrý nápad, ale není to přesně to, co bychom chtěli. Necháváme za sebou zlatou zídku, což ale neodpovídá představě chodníku. Jak bychom mohli zapustit zlaté bloky do země tak, aby z nich vznikal chodník?

Změňme souřadnici Y tak, že od ní odečteme jedničku, aby se bloky umisťovaly o jednu úroveň dolů.

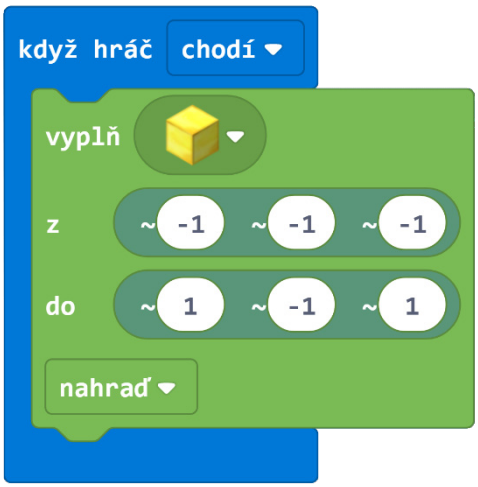


A teď se podívejme, co se stane:

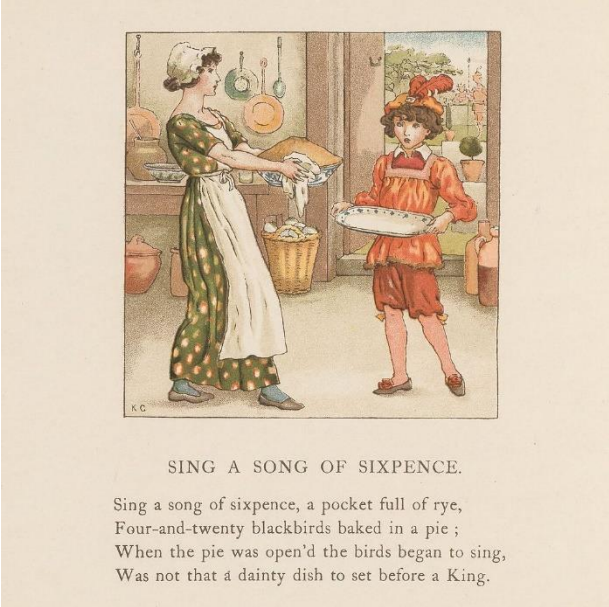


Tak to je mnohem lepší! Jako vylepšení můžete vytvářet celou zlatou cestu širokou 3 bloky. Budete tak moci po cestě procházet s přítelem vedle sebe. Nápověda: použijte jiný blok z nabídky Bloky!

Řešení:



Aktivita: Zazpívejte píseň o šestáku



"Sing a song of sixpence,
A pocket full of rye.
Four and twenty blackbirds,
Baked in a pie." – [English Nursery Rhyme]

V této aktivitě se žáci pokusí znázornit starou anglickou říkanku, v které se zjednodušeně píše, že když se rozlomí koláč, vyletí z něj 24 kosů.

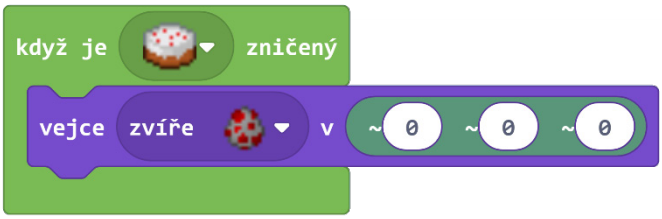
Místo kosů použijeme papoušky a místo koláče dort!

Kroky:

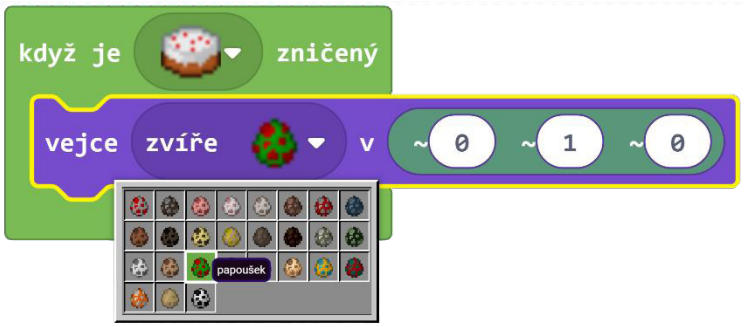
1. Z nabídky Bloky přetáhněte blok Když je zničený do pracovního prostoru. To bude náš zachytávač událostí.

2. Použijte rozbalovací seznam a vyberte položku dort.

3. Z nabídky Stvoření připojte blok Vejce v dovnitř do bloku Když je zničený. Ujistěte se, že je správně připojený.

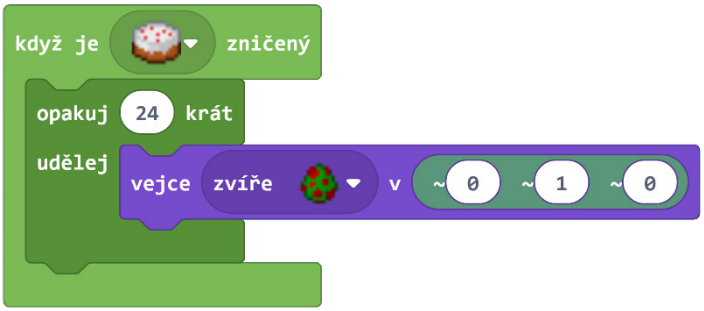


4. Použijte rozbalovací seznam v bloku Vejce a vyberte papoušek.
5. Chceme, aby se papoušek objevoval nad naší hlavou, proto změníme v bloku Vejce souřadnici Y na 1.



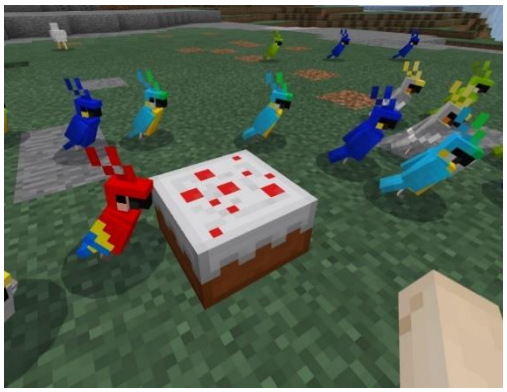
Tak se nám objeví nad hlavou právě jeden papoušek, použijeme tedy smyčku Opakuj, abychom nechali objevit všech 24 papoušků.

6. Z nabídky Smyčky přetáhněte blok Opakuj přímo do bloku Když je zničený a poté do něj vložte náš blok Vejce pro vytvoření 1 papouška.
7. V nastavení smyčky Opakuj napište číslo 24.



Abyste spustili tento náš program ve hře, vložte do inventáře hráče dort (pro zpřístupnění inventáře stiskněte klávesu ‘E’). Položte dort na zem tak, že ho vyberete ve vaší nabídce nástrojů a klikněte pravým tlačítkem myši někam na zem.

Pak už jen stačí pomocí levého tlačítka myši do dortu udeřit, abyste dort rozlomili – mělo by se objevit hejno papoušků!



Celý program naleznete zde: https://makecode.com/_0ji3UvTDg4Ds

Aktivita: Nepřátel se nelekejme, na množství nehleďme

V této aktivitě si vytvoříme situaci, kdy budeme napadeni hordami zombie. Ukážeme si efekt exponenciálního růstu: pokaždé, když zabijete jednoho zombie, objeví se na jeho místě dva další. Jak se cítíte? Pojďme se podívat na tento hororový svět!

Než začnete, ověřte si, že jste na bezpečném místě nebo v místě, které se dobře brání. Stromy v džungli jsou na to skvělé, tak jako věže hradu nebo někde za ohradou (pokud se nacházíte uvnitř). V případě, že se nacházíte na plochém světě, můžete utíkat a střílet zombie za běhu. Je to situace, kdy nelze vyhrát, tak s tím počítejte a užijte si to.

Zachytávač událostí, který použijeme k vytvoření nových „zombíků“ v situaci, kdy je jeden z nich zabit, je „Když je zvíře zabitý/á/é.“

Kroky:

1. Z nabídky Stvoření vyberte blok Když je zabitý/á/é a přetáhněte ho na pracovní plochu.

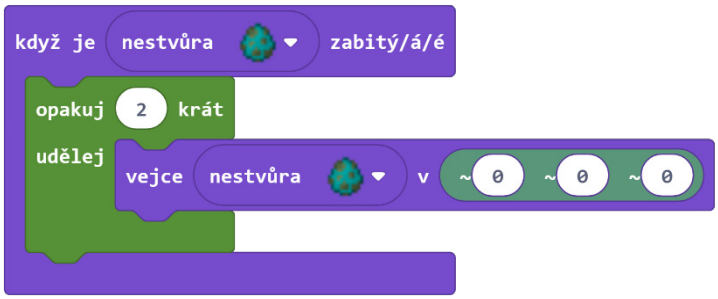
2. Z nabídky Stvoření přetáhněte na pracovní plochu blok Nestvůra dovnitř do bloku Když je zabitý/á/é a nahraďte jím přednastavený blok představující zvíře.
3. Použijte rozbalovací seznam v bloku Nestvůra a vyberte vejce pro vytvoření zombie.

Teď se musíme postarat o to, aby se v situaci, kdy je zombie zabit, objevili dva noví na náhodném místě poblíž hráče.

4. Přetáhněte z nabídky Stvoření blok Vejce dovnitř do bloku zachytávače události Když je zabitý/á/é.

5. Z nabídky Stvoření přetáhněte blok Nestvůra dovnitř do bloku Vejce.
6. Použijte rozbalovací seznam v bloku Nestvůra a vyberte vejce pro vytvoření zombie.

7. Protože chceme, aby se při zabití jednoho zombie objevili dva noví, vyberte z nabídky Smyčky blok Opakuj a vložte ho dovnitř do bloku Když je zabitý/á/é. Poté přetáhněte blok Vejce dovnitř do bloku smyčky.
8. Změňte parametr počtu opakování smyčky Opakuj na 2.

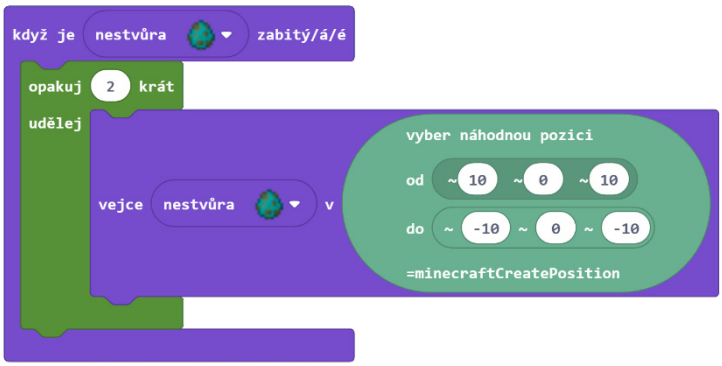


Teď nám zbývá říct Minecraftu, kde se mají noví zombie objevit. Normálně by se objevili v místě, kde se přímo nachází hráč. Tedy na relativních souřadnicích (~0 ~0 ~0). Nechat zombie, aby se objevili přímo na místě, kde se nacházíte, není dobrý nápad. Pojďme se tedy postarat o to, aby se objevili trochu dále od nás na náhodném místě a dali nám tím šanci zareagovat na případnou obranu.

9. Z nabídky Pozice přetáhněte blok Vyber náhodnou pozici a umístěte ho dovnitř do bloku Vejce tak, že nahradíte výchozí souřadnice.

Dvě skupiny souřadnic zadávaných v bloku Vyber náhodnou pozici popisují protilehlé rohy prostoru ve tvaru kvádru, ve kterém se na náhodném místě objeví zombie. Zvolme oblast deseti bloků kolem našeho hráče. Tím vytvoříme oblast o velikosti 400 čtverečních bloků kolem místa, na kterém náš hráč stojí.

10. V bloku Vyber náhodnou pozici nastavte souřadnice *od* na (~10 ~0 ~10) a souřadnice *do* na (~-10 ~0 ~-10)



JavaScript:

```
mobs.onMobKilled(mobs.monster(ZOMBIE), function () {
    for (let index = 0; index < 2; index++) {
        mobs.spawn(mobs.monster(ZOMBIE), randpos(
        pos(10, 0, 10),
        pos(-10, 0, -10)
        ))
    }
})
```

Sdílený program: https://makecode.com/_VseXqc2Vjbv9

- Abyste si mohli tento strašidelný zážitek užít, otevřete svět Minecraftu v Kreativním módu.
- Otevřete svůj inventář a vybavte se zbraněmi (ideálně zbraněmi s dalekým dostřelem jako je luk). A také si vezměte vejce zombie (můžete ho nalézt pomocí vyhledávání).

- Přepněte hru do režimu Přežití – v menu Nastavení nebo napsáním příkazu v chatovacím okně: „`/gamemode s`.“

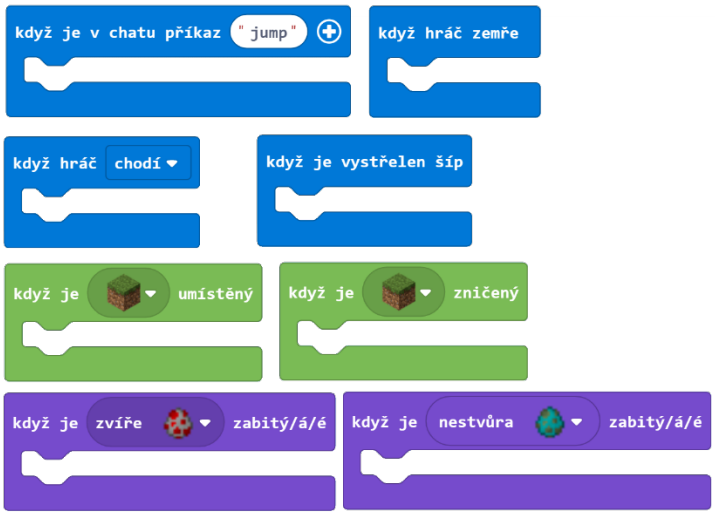
- Nastavte ve hře čas na noc. Zombie za světla umírají samy. My ale chceme, aby umíraly až poté, co je sami zabijeme. Napište v chatovacím okně příkaz: „`/time set night`.“
- Tip – ujistěte se, že nemáte nastavenou obtížnost hry na Mírumilovná – to by na vás totiž „zombíci“ neútočili, resp. byste je ani nevytvořili (obtížnost zjistíte v nastavení světa).
- Nakonec se skryjte ve vašem úkrytu, hoďte vejce Zombie mimo vaše teritorium a teror může začít!

Samostatný projekt: Události

Poznámka pro učitele:

Kreativní projekty s otevřeným koncem jsou skvělou příležitostí k procvičení probrané kapitoly. V Minecraftu je celá řada problémů, které můžete řešit: Jak udržíte krávy pohromadě, aby vám neutekly? Jak vytvořit obnovitelný zdroj vody? Jak se dostat zpátky na povrch, když spadnete do rokle? Projekty umožňují žákům řešit problémy, které jsou pro ně smysluplné. Pomocí MakeCode mohou žáci navrhnout originální, automatizované řešení k mnoha problémům, kterým čelí v jejich každodenním „minecraftím“ (reálném) životě. Je důležité dát žákům příležitost vytvořit originální projekty, aby prokázali, že mohou aplikovat své nové dovednosti zcela novým způsobem a trénovat svou kreativitu.

Zopakujme si, co jsme se naučili v této kapitole. V průběhu hry se neustále odehrávají nějaké události. Abyste s těmito událostmi mohli pracovat – reagovat na ně – potřebujete vytvořit a naplnit blok zachytávače události. V tomto projektu je na žácích, aby sami vymysleli systém reakcí na události pomocí MakeCodu pro Minecraft tak, aby vytvořili celistvý projekt. K tomu mohou použít jeden nebo více zachytávačů událostí s využitím těchto bloků:



Bloky zachytávačů událostí:

- Když je v chatu příkaz
- Když hráč zemře
- Když hráč chodí/jezdí/padá/plave ve vodě/ a další.
- Když je vystřelen šíp
- Když je blok umístěný
- Když je blok zničený
- Když je zvíře zabité
- Když je nestvůra zabitá

U některých typů těchto bloků máte mnoho možností pro výběr typu bloku, události, akce nebo živočicha.

Příklady:

1. Kaleidoskop

V tomto projektu se pokaždé, když je umístěna kamenná kostka, umístí další 3 symetricky k vaší pozici.

X	O
O	O

- X = umístěný blok
- O = nové bloky

2. Chůze po vodě

Když jdete po vodě jezera, řeky nebo moře, umisťujete bloky ledu nebo skla pod své nohy, takže to vypadá, jako byste kráčeli po vodě.

3. Pán počasí

Vytvořte příkazy pro chatovací okno, které spustí déšť nebo naopak zajistí perfektní počasí.

4. Když já, tak všichni

Vytvořte reakci, která se postará o to, že všichni ve hře zemřou, pokud zemře váš hráč.

Minecraft deník

Zapište si do deníku:

- Jaký druh události a reakce na ni jste se rozhodli použít?
- Co má program dělat? Popište, jak má program pracovat (co je akce a co je reakcí na ni).
- Vložte do deníku alespoň jeden snímek obrazovky, abyste znázornili, jak program pracuje.
- Sdílejte svůj projekt na webu a do deníku vložte URL odkaz.

POZNÁMKA:

Pokud jste se rozhodli vylepšit některou z aktivit v této lekci, promluvte si o tomto novém kódu, který jste napsali. A vysvětlete si toto vylepšení.

Bodové hodnocení

	1	2	3	4
Deník	V deníku chybí 4 nebo více požadovaných bodů.	V deníku chybí 2 nebo 3 z požadovaných bodů.	V deníku chybí 1 z požadovaných bodů.	V deníku jsou zpracovány všechny požadované body.
Projekt	V projektu chybí všechny požadované součásti.	Projekt používá alespoň jeden blok zachytávače událostí nebo řeší nějaký problém, ALE kód je neefektivní nebo chybný.	Projekt používá alespoň jeden blok zachytávače událostí; NEBO řeší problém.	Projekt používá alespoň jeden blok zachytávače událostí; A ZÁROVEŇ s jeho pomocí působí zamýšlený efekt nebo řeší problém.

Souřadnice

V této lekci si ukážeme, jak se orientovat ve světě Minecraftu v rámci jeho trojrozměrného souřadného systému reprezentovaného trojicemi souřadnic (X Y Z) a vysvětlíme si rozdíly mezi relativními souřadnicemi a přesným určením místa ve světě (absolutními souřadnicemi).

Také si ukážeme, jak si postavit ukazatel světových stran, abychom znali směr našich cest. Vytvoříme si nástroj, který nám umožní vytvářet kopie struktur v Minecraftu pomocí "Kopírovat" a "Vložit", tak jak jsme běžně zvyklí se soubory v počítači.

Obrázek 1: Souřadnice ve světě Minecraftu – X, Y a Z !

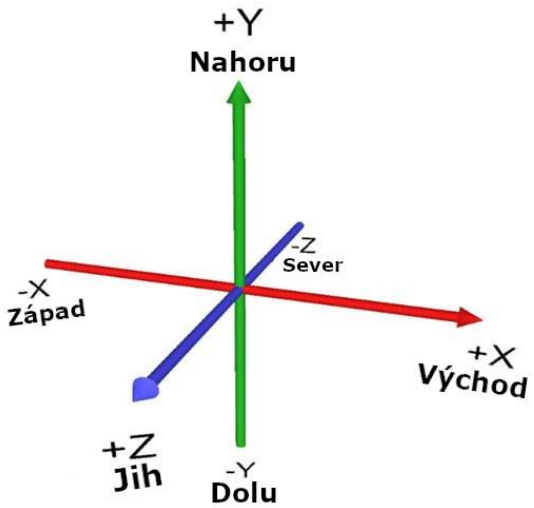
Proč studovat souřadnice?

V Minecraftu je velmi důležité znát svou pozici a pozici, kde se nachází váš Agent. Důležité jsou i informace, kde se nachází diamantové doly, lesy nebo třeba podzemní místnosti, kde se objevují příšery. V módu „Přežití“ je pro nás životně důležité, abychom byli při západu slunce schopni nalézt naše útočiště nebo vyhledat místo, kde máme truhlu, abychom si mohli aktualizovat náš inventář, když už ho máme přeplněný, nebo si vzít nový nástroj místo zničeného.

Tato lekce vám může pomoci, i když už třeba nejste v Minecraftu žádní nováčci a pohybujete se v něm se šarmem prospektorů a budovatelů velkoměst. Pokud totiž chcete používat bloky v MakeCode opravdu efektivně, je nezbytné pochopit jejich systém. *Souřadnice* nám totiž vždy určují *pozici*.

Souřadnice v Minecraftu

Minecraft používá k určení konkrétního místa systém 3 hodnot (X, Y, Z), tzv. souřadnic. Tento princip používá také MakeCode pro Minecraft, aby určoval místo nebo prostor, kde má proběhnout nějaká akce.



Tak jako v každé normální souřadné soustavě, kterou používáme například ve fyzice nebo v matematice, je i v „minecraftím“ souřadném systému *počátek*, tedy místo se souřadnicemi (0, 0, 0). Všechny souřadnice popisují vzdálenost od tohoto bodu v konkrétním směru.

- Souřadnice X reprezentuje horizontální vzdálenost ve směru východ-západ. V našem běžném světě odpovídá *zeměpisné délce*.
    - Směrem na východ nabývá souřadnice X kladných hodnot (+X).
    - Směrem na západ nabývá souřadnice X záporných hodnot (-X).
- Souřadnice Y představuje vzdálenost ve směru nahoru a dolů. Tento význam odpovídá v reálném světě *nadmořské výšce*.
    - Směrem nahoru hodnoty souřadnice Y rostou (+Y).
    - Směrem dolů hodnoty souřadnice Y klesají (-Y).
- Souřadnice Z popisuje vzdálenosti ve směru sever-jih od počátku. V reálném světě představuje *zeměpisnou délku*.
    - Směrem na sever roste hodnota souřadnice Z (+Z).
    - Směrem na jih naopak klesá hodnota souřadnice Z (-Z).

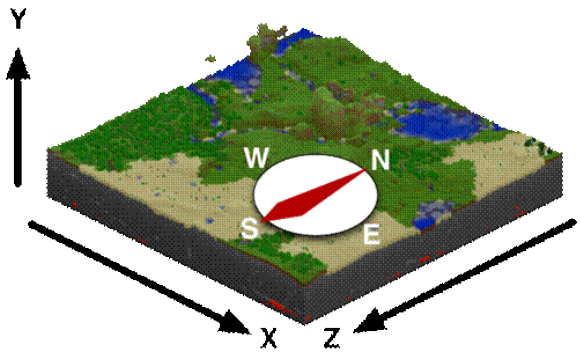
V levém horním rohu se zobrazí souřadnice vaší pozice (X, Y, Z). Pokud souřadnice nevidíte, přejděte do nastavení a tuto možnost aktivujte.

Na výše uvedeném obrázku vidíte světové souřadnice (52, 77, -71). Tato čísla představují, jak daleko se nacházíte od počátku světa (0, 0, 0):

- X = 52 znamená, že jste 52 bloků východně od počátku
- Y = 77 znamená, že jste 77 bloků nad místem počátku (počátek je v oblasti bedrocku – podloží, nejníže položený kámen světa)
- Z = -77 znamená, že jste 71 bloků jižně od počátku

Zkuste procházet světem Minecraftu a sledujte, jak se vám mění zobrazené souřadnice. Když zmáčknete mezerník a vyskočíte, uvidíte, že se změnila naše poloha směrem nahoru, tedy souřadnice Y se změnila o +1.

Absolutní světové souřadnice vs Relativní pozice hráče



Ve hře Minecraft je potřeba uvažovat o dvou pojetích chápání místa:

- Absolutní světové souřadnice ve světě
- Relativní vzdálenost od nějakého jiného místa nebo od mé pozice

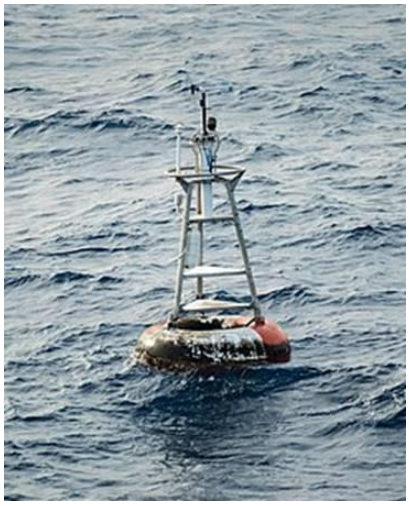
Absolutní světové souřadnice

Absolutní světové souřadnice jsou složeny ze 3 čísel, která představují vzdálenost od počátku „minecraftího“ světa (0, 0, 0). Tak jako v našem reálném světě – pozice na zeměkouli jsou pro dané místo pořád stejné a NEZMĚNÍ se v průběhu hry. Jedno místo má pořád stejné souřadnice a nezáleží, kde zrovna stojí hráč ve světě.

Příklady z reálného světa:

- "Počátek" souřadnic na zeměkouli se nachází na rovníku a na Greenwichském nultém poledníku – zeměpisná délka 0 i šířka 0. Tento bod je označený meteorologickou bójí u pobřeží Afriky.

- Empire State Building má světové souřadnice 40 stupňů severní šířky a 73 stupňů západní délky (vzhledem k průsečíku Greenwichského poledníku a rovníku), což je možné také zapsat jako 40°N -73°E.
- Mount Everest má zeměpisné souřadnice 27 stupňů severní šířky a 86 stupňů východní délky od počátku.
- Pražský hrad má zeměpisné souřadnice 50°N 14°E



Meteorologická bóje - 0°N 0°E (fiktivní Null Island)

Empire State Building - 40°N - 73°E

Mount Everest - 27°N 86°E

Nezáleží na tom, kde zrovna jste nebo kde se právě pohybujete, světové souřadnice Mount Everestu nebo Pražského hradu zůstanou pořád stejné.

Jejich relativní souřadnice se měnit mohou, ale jejich absolutní souřadnice jsou stále stejné.

Relativní pozice od hráče

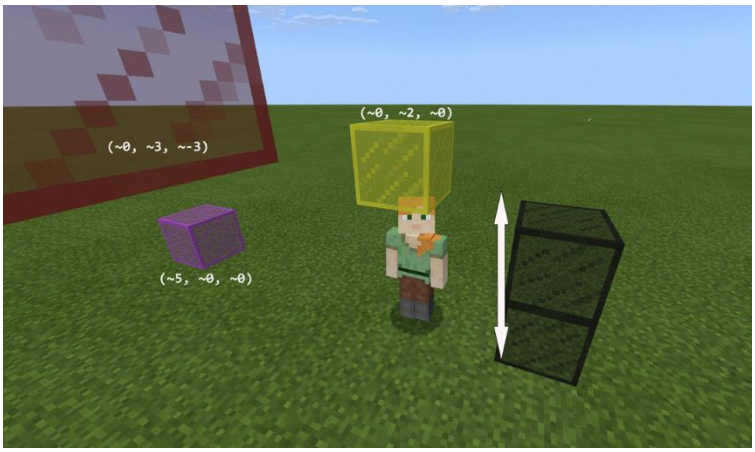
Relativní pozice je odvozena od místa, kde se právě nachází hráč. Tedy počátkem relativních souřadnic je místo, kde hráč právě stojí. Relativní souřadnice jsou vždy předznamenány symbolem '~' (vlnovka) před každým ze 3 čísel souřadného systému světa Minecraftu (~X, ~Y, ~Z).

Pozice představuje prostor, který je velký jako jedna kostka o velikosti 1 x 1 x 1 bloků světa Minecraftu. Hráč sám o sobě je vysoký 2 bloky. Souřadnice (~0 ~0 ~0) představují hráčovy nohy a (~0 ~1 ~0) představují hráčovu hlavu.

Tak, jak se hráč pohybuje po světě Minecraftu, hýbou se i nohy hráče, a proto se pohybuje počátek relativních souřadnic. Takže relativní souřadnice NEJSOU pevné – mění se neustále podle toho, jak se hráč pohybuje.

Například:

- (~0, ~2, ~0) představuje místo přímo nad hráčovou hlavou (žlutá kostka)
- (~5, ~0, ~0) je blok umístěný na zemi 5 bloků na východ od hráče (fialová kostka)
- (~0, ~3, ~-3) je blok v místě 3 bloky vysoko a 3 bloky severně od místa, kde hráč stojí (červená kostka).



Offline: Světové značky

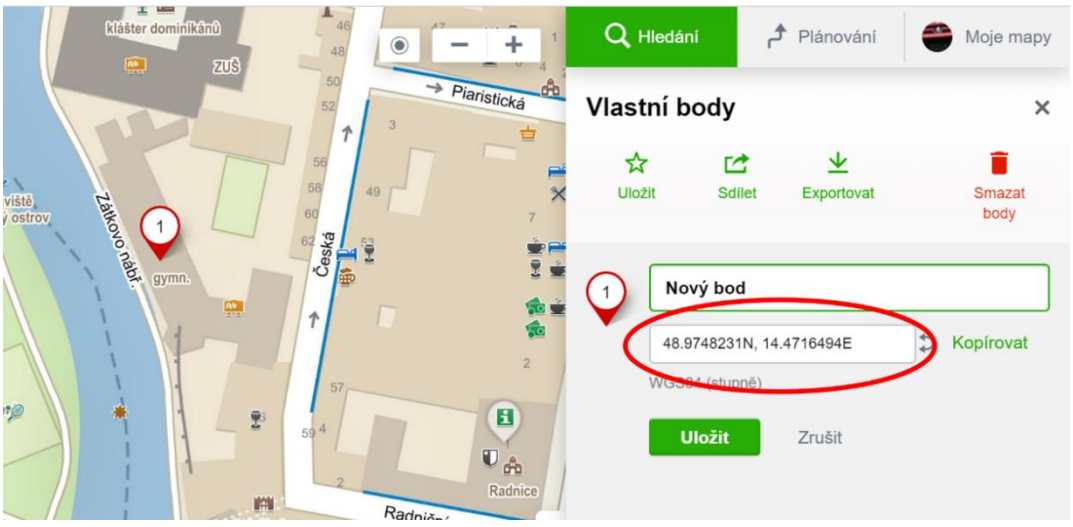
V této aktivitě se naučíte pracovat s relativními a absolutními souřadnicemi a procvičíte si rozdíly mezi nimi.

Potřebné materiály:

- Glóbus nebo mapa se značením zeměpisné šířky a délky
- Nebo mapovací/GPS software - Pokud máte přístup k mapě světa nebo glóbu, můžete najít svou vlastní pozici v reálném světě (kde ve světě stojíte, vaše zeměpisná šířka a délka) nebo kde jsou různé památky, jako Empire State Building nebo Velká pyramida v Gíze ve vztahu k vaší současné pozici.

Poznámky:

- V závislosti na možnosti přístupu k mapám, glóbům nebo aplikacím GPS a na věku žáků můžete požadovat, aby žáci použili místo své vlastní pozice hlavní město své země. Také můžete upravit, jak přesní budete v zaznamenávání zeměpisné délky a šířky. Zde jsme využili základní stupně bez podrobnějších minut a vteřin.
- Většina mapovacích softwarů, jako například Seznam Mapy mají funkci "Moje lokace", která vám sdělí vaše aktuální souřadnice.



- Na Zemi je zeměpisná šířka na rovníku 0 stupňů a na severním pólu je 90 stupňů severně. Jeden stupeň zeměpisné šířky pokrývá zhruba 112 kilometrů.

Kroky:

1. Najděte svou vlastní pozici v reálném světě, zeměpisnou šířku a délku místa, kde právě jste.
2. Vyberte si světový orientační bod a vyhledejte souřadnice tohoto bodu.
3. Vypočítejte pozici tohoto bodu ve vztahu ke své vlastní současné pozici.

Například, kdybyste právě byli na věži Space Needle v Seattlu, Washington:

- Vaše světová pozice by byla 47 stupňů severní šířky a 122 stupňů západní délky.
- Vyberte si věž Empire State Building jako svůj orientační bod. Světové souřadnice Empire State Building jsou 40 stupňů severní šířky a 73 stupňů západní délky.
- Vypočítávání pozice Empire State Building relativně k vaší současné pozici:
    - Moje pozice (47, 122)
    - Empire State Building (40, 73)
    - Relativní pozice Empire State Building vzhledem ke mně (47-40=7, 122-73=49) takže (~7, ~49)

Empire State Building by byla 7 stupňů jižně a 49 stupňů východně od vás. Pamatujte, že tyto relativní pozice mají "~" (vlnku) před každým číslem, zatímco absolutní světové pozice ji nemají.

Offline: Třídní souřadnice

Procvičte si rozdíl mezi relativními souřadnicemi a absolutními souřadnicemi ve vaší vlastní třídě.

Jak obecné nebo detailní souřadnice vytvoříte, závisí na věku a schopnostech vašich žáků.

Potřebné materiály:

- Maskovací páska (volitelné)
- Indexové karty nebo značky
- Čtverečkovaný papír (volitelné)

Příprava: Identifikujte, který směr nebo stěna ve vaší třídě reprezentuje jeden ze základních směrů: sever, jih, východ, západ a podle toho je označte. Pokud možno, nalepte pásku na zem, aby reprezentovaly osy X a Z pro vytvoření souřadnicové mřížky se začátkem ve středu třídy, kde se osy setkávají. Toto funguje nejlépe, pokud máte na zemi dlaždice nebo jiný čtvercový vzor. Připomeňte žákům, že osa Y je svislá osa shora dolů.

Absolutní pozice žáků:

1. Představte si, že vaše třída je samostatný Minecraft svět.
2. Položte počáteční bod a cedulku značící světovou pozici (0, 0, 0).
3. Postavte se na tento počáteční bod.
4. Požádejte žáky, ať napíší na indexovou kartu nebo papír svou pozici ve třídě ve formátu (X, Y, Z). Jinými slovy, jak byste museli putovat rovnoběžně s osami tak, abyste se dostali na svou pozici? (Pokud na podlaze nemáte vzor, považujte za jeden blok jeden krok) Pro toto cvičení je nechte na souřadnici (0) na ose Y.
5. Zkontrolujte pozice několika žáků tím, že půjdete od nulového bodu uprostřed třídy ke každému žáku, abyste se ujistili, že mají pravdu.

Příklad: Žák dá své třídě souřadnice (4, 0, 2). Pokud půjdeš 4 kroky východně a 2 kroky jižně ze středu třídy, měl bys dojít na jeho pozici.

K diskusi:

- Co určuje průměrný krok?
- Jaká je hranice chyby pro malé vzdálenosti?
- Co dlouhé vzdálenosti?
- Proč není tento problém v Minecraftu?

Absolutní pozice předmětů:

1. Vyberte objekt z pokoje, který má trvalé místo.
2. Řekněte žákům, ať identifikují a označí pozici tohoto objektu ve třídě.
3. Pokud je objekt nad zemí, zadejte jim, ať zadají souřadnici Y jako jinou než nula (Budete potřebovat neměnnou svislou míru, např. 30 cm = 1 blok).

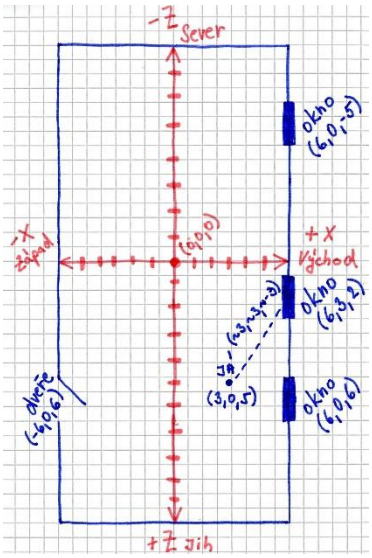
Opakujte předchozí kroky s dalšími objekty, které mají stálé místo ve třídě – například nástěnné hodiny, dveře nebo okno. Řekněte žákům, ať označí tyto předměty jejich příslušnými světovými souřadnicemi. Když máte pocit, že žáci zvládají koncept absolutní světové pozice, posuňte se k další sekci.

Relativní pozice studentů:

1. Připomeňte žákům, že při používání relativní pozice je počáteční bod u nich (~0 ~0 ~0).
2. Vyberte předmět ve třídě, který je popsán svou absolutní světovou pozicí z předchozího cvičení.
3. Požádejte žáky, aby zapsali na indexovou kartu nebo papír, jak daleko a kterým obecným směrem (směry) by museli putovat ze svého místa, aby se k předmětu dostali. Měli by použít formát (~X ~Y ~Z).
4. Řekněte žákům, aby se přemístili po třídě a znovu vypočítali pozici objektu ve vztahu k jejich nynější pozici. Opět by měli použít formát (~X ~Y ~Z).
5. Opakujte předchozí krok s dalšími předměty, které mají ve třídě neměnné místo.

Poznámky:

- Můžete navrhnout, ať žáci "zmapují" třídu a lokaci svou a předmětů na čtverečkovaný papír.
- Zkontrolujte relativní pozice žáků tím, že přejdou ze svého místa k předmětu podle souřadnic, které si z tohoto místa napsali.
- Například, žák může říct, že předmět je na pozici (~-2 ~0 ~-5) vzhledem k jeho pozici. Musel by tedy putovat 2 dlaždice na západ a 5 dlaždic na sever, aby se k objektu dostal.



Upozorněte na rozdíly mezi absolutní pozicí a relativní pozicí:

- Absolutní pozice objektu s neměnným místem se NEMĚNÍ.
- Pozice objektu relativně k pozici žáka se mění, když se žák po třídě pohybuje.

Dodatečné úlohy: Relativní pozice pohyblivých předmětů

- Rozdělte žáky do dvojic, aby počítali relativní pozici partnera ve vztahu k sobě samému.
- Řekněte jim, ať se pohybují a přepočítávají své pozice relativně ke své vlastní.

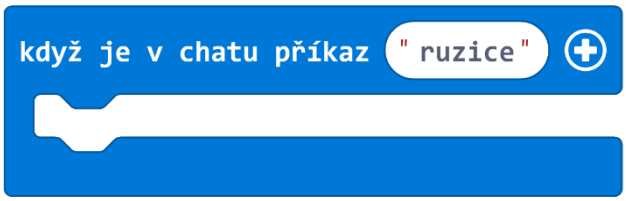
Aktivita: Vytvořte si směrovou růžici

Žáci budou procvičovat použití souřadnic v Minecraftu, aby vytvořili směrovou růžici, která míří do čtyř hlavních směrů v Minecraftu.

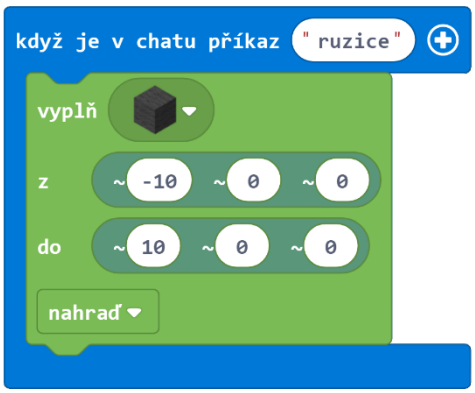
Poté ji mohou použít jako základ pro vytváření samostatných a detailnějších růžic.

Kroky:

1. Vytvořte nový projekt v MakeCode nazvaný "ruzice".
2. Přetáhněte blok Když je v chatu příkaz z nabídky Hráč na panelu nástrojů a přepište jej na „ruzice“.



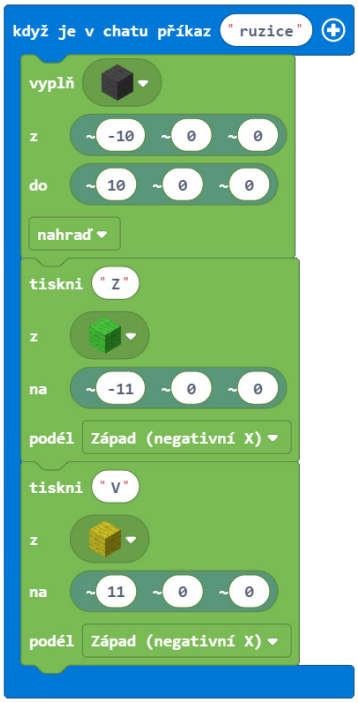
3. Jděte do nabídky Bloky a přetáhněte blok Vyplň dovnitř Když je v chatu příkaz.
4. Pomocí rozbalovací nabídky v bloku Vyplň vyberte blok, z kterého bude osa vaší růžice postavená.
5. Zadejte hodnoty (~-10, ~0, ~0) do pole od a (~10, ~0, ~0) do pole do (~10, ~0, ~0), abyste vystavěli 20 bloků podél osy X.



V nabídce Bloky na panelu nástrojů najděte blok Tiskni. Tento blok vytiskne slova z jakéhokoli bloku vašeho výběru podél zadané osy.

6. Vytáhněte dva tyto bloky Tiskni a položte je za blok Vyplň dovnitř bloku Když je v chatu příkaz.
7. Zadejte „Z“ do prvního Tiskni bloku a „V“ do druhého Tiskni bloku, abyste si označili západ a východ.
8. Pomocí rozbalovací nabídky v bloku Tiskni vyberte blok, ze kterého budou tato písmena postavená.
9. V prvním Tiskni bloku „Z“ zadejte hodnoty (~-11, ~0, ~0).

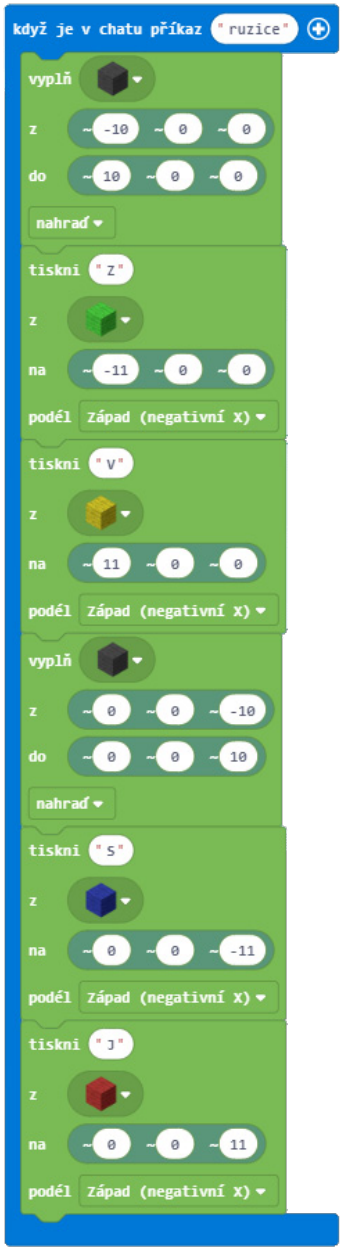
10. V druhém Tiskni bloku „V“ zadejte hodnoty (~11, ~0, ~0).



Opakujte kroky 3–10, ale tentokrát pro severo-jižní osu směrové růžice:

11. Z nabídky Bloky na panelu nástrojů přetáhněte blok Vyplň dovnitř Když je v chatu příkaz.
12. Pomocí rozbalovací nabídky v bloku Vyplň vyberte blok, ze kterého chcete mít tuto osu růžice.
13. Zadejte hodnoty (~0, ~0, ~-10) do pole od a hodnoty (~0, ~0, ~10) do pole do, abyste si vytvořili čáru z 20 bloků podél osy Z.
14. Vytáhněte dva Tiskni bloky a vložte je za blok Vyplň do Když je v chatu příkaz.
15. Zadejte „S“ do prvního bloku Tiskni a „J“ do druhého bloku Tiskni, abyste označili sever a jih.
16. Pomocí rozbalovací nabídky v Tiskni bloku vyberte blok, kterým chcete tato písmena vypsat.
17. V prvním Tiskni bloku „S“ zadejte hodnoty (~0, ~0, ~-11).
18. V druhém Tiskni bloku „J“ zadejte hodnoty (~0, ~0, ~11).

Váš kompletní program by měl vypadat takto:



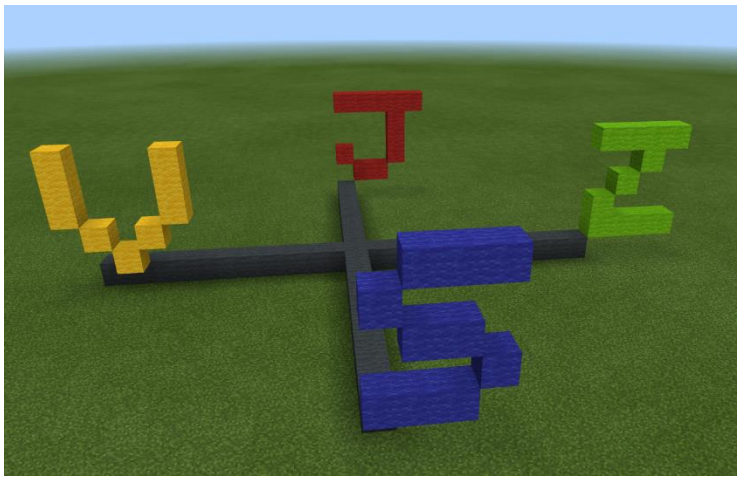
JavaScript:

```
player.onChat("ruzice", function () {
    blocks.fill(
    GRAY_WOOL,
    pos(-10, 0, 0),
    pos(10, 0, 0),
    FillOperation.Replace
    )
    blocks.print(
    "Z",
```

```
    LIME_WOOL,
    pos(-11, 0, 0),
    WEST
    )
    blocks.print(
    "V",
    YELLOW_WOOL,
    pos(11, 0, 0),
    WEST
    )
    blocks.fill(
    GRAY_WOOL,
    pos(0, 0, -10),
    pos(0, 0, 10),
    FillOperation.Replace
    )
    blocks.print(
    "S",
    BLUE_WOOL,
    pos(0, 0, -11),
    WEST
    )
    blocks.print(
    "J",
    RED_WOOL,
    pos(0, 0, 11),
    WEST
    )
})
```

Sdílený program: https://makecode.com/_P4bCDV4CR7WT

Běžte do plochého světa v Minecraftu a zadejte „ruzice“ do příkazové řádky.
Pro lepší výsledek stůjte nehybně po celou dobu tvoření písmen – pamatujte, že pozice se vypočítávají ve vztahu k vám jako hráči, takže pokud se budete hýbat, písmena se objeví na různých místech!



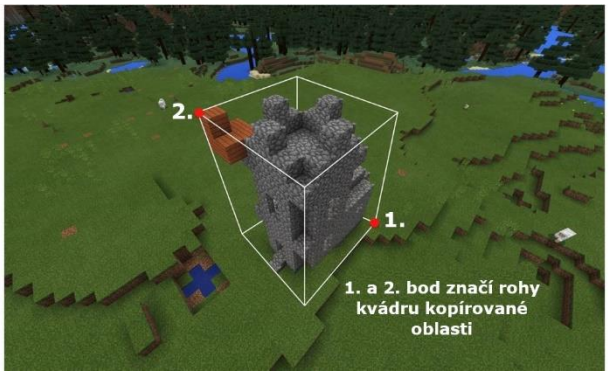
Dodatečné úkoly k projektu směrové růžice:

- Přidejte označení nahoru a dolů po ose Y.
- Vytvořte svůj kompas ve vzduchu, takže bude vidět z jakékoliv pozice ve světě.
- Prodlužte osu svého kompasu na více než 20 bloků.
- Místo bloků Vyplň a Tiskni naprogramujte svého Agenta, aby pro vás vytvořil růžici.
- Využívejte Stavitel bloky k vytvoření růžice.

Další referenční příklady:

- Tutoriál na směrovou růžici – https://minecraft.makecode.com/tutorials/compass-rose
- Příklad 3D osy – https://minecraft.makecode.com/examples/3d-axis
- Příklad kompasu – https://minecraft.makecode.com/examples/compass

Aktivita: Stěhovací firma Minecraft



Zažili jste v Minecraftu situaci, kdy jste postavili dokonalý dům, a poté zjistili, že byste ho chtěli přesunout na jinou pozici? Třeba k moři? Naučíme se, jak na to! Můžete použít souřadnice a některé bloky z nabídky Bloky pro kopírování a vkládání celých částí prostředí v Minecraftu. Můžete kopírovat svahy, jezera nebo celé budovy! Pojďme vytvořit postup, jak označit oblast světa a zkopírovat ji někam jinam.

V tomto projektu využijeme 3 různé bloky Když je v chatu příkaz:

- „start“: Nastaví jeden roh kopírované oblasti.
- „stop“: Nastaví opačný roh kopírované oblasti.
- „kopirovat“: Vytvoří přesnou kopii všeho mezi počátečním a konečným bodem na vaší současné pozici.

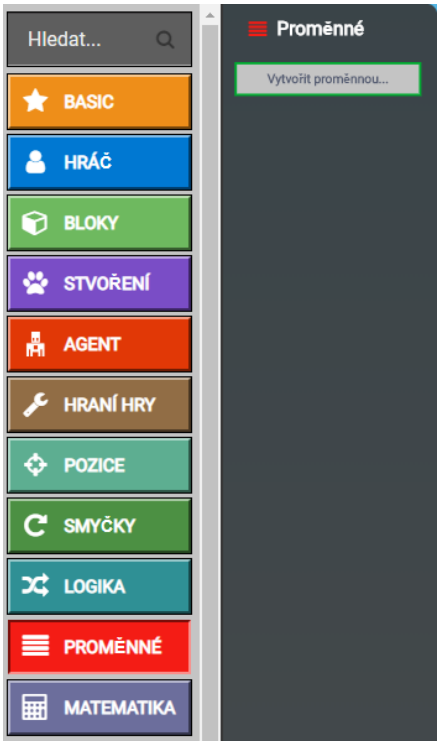
Kroky:

1. Vytvořte nový MakeCode projekt s názvem „kopirovat“.
2. Následně přetáhněte tři Když je v chatu příkaz bloky do pracovního prostoru.

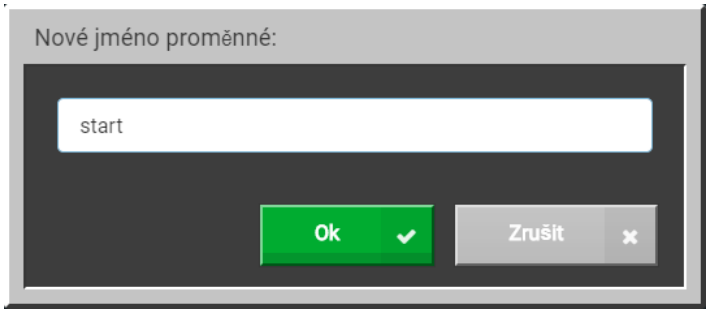
3. Přejmenujte tyto bloky na „start“, „stop“ a „kopirovat“.

Vytvoříme několik proměnných, které využijeme k ukládání počátečních a konečných pozic. O proměnných budeme víc mluvit v samostatné kapitole, ale zatím si jen pamatujte, že jedna proměnná může reprezentovat tři samostatné souřadnice, které popisují absolutní světovou pozici.

4. Otevřete nabídku Proměnné na panelu nástrojů a klikněte na možnost Vytvořit proměnnou.



5. Pojmenujte tuto proměnnou *start* a klikněte na Ok.



6. Opět klikněte na tlačítko Vytvořit proměnnou.
7. Pojmenujte ji *stop*, a klikněte na Ok.

Teď nastavme hodnotu těchto proměnných v závislosti na současné světové pozici hráče.

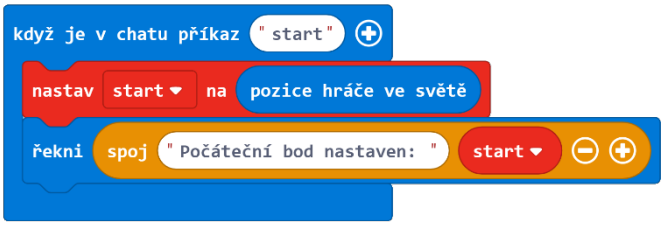
8. Z karty Proměnné na panelu nástrojů vyberte blok proměnné Nastav a přesuňte jej dovnitř bloku Když je v chatu příkaz „start“.
9. Pomocí rozbalovací nabídky v Nastav bloku vyberte proměnnou *start.*
10. Z nabídky Hráč přetáhněte blok Pozice hráče ve světě dovnitř bloku Nastav start místo 0.

Teď vypišme informační zprávu, která nahlásí souřadnice, které jsme uložili jako počáteční pozici.

11. Z nabídky Hráč přetáhněte blok Řekni za blok Nastav start.

12. Klikněte na tlačítko Rozšířené v panelu nástrojů, abyste zobrazili pokročilé kategorie.

13. Z nabídky Text přetáhněte blok Spoj do bloku Řekni namísto „:)“.
14. Do prvního políčka bloku Spoj napište „Počáteční bod nastaven:“.
15. Z nabídky Proměnné na panelu nástrojů přetáhněte blok proměnné Start do druhého políčka bloku Spoj.



Opakujte kroky 4-15, ale pro konečnou pozici. Nápověda, můžete kliknout pravým tlačítkem myši na jakýkoli blok a vybrat možnost Klonovat, abyste blok zkopírovali. Toto je dobrá strategie, když chcete ušetřit trochu času při tvoření větších bloků podobného kódu!

16. Z nabídky Proměnné v panelu nástrojů přetáhněte blok Nastav dovnitř bloku Když je v chatu příkaz „stop“.
17. Z rozbalovací nabídky v bloku Nastav vyberte proměnnou *stop.*
18. Z nabídky Hráč v panelu nástrojů přesuňte blok Pozice hráče ve světě do bloku Nastav stop místo 0.
19. Z nabídky Hráč přetáhněte blok Řekni za blok Nastav stop.
20. Z nabídky Text přetáhněte blok Spoj do bloku Řekni místo „:)“.
21. V prvním bloku Spoj napište „Koncový bod nastaven:“.
22. Z karty Proměnné přetáhněte blok Stop do druhého políčka bloku Spoj.

Teď, když máme začáteční a konečné body, vytvořme možnost kopírovat danou oblast mezi body.

23. Z nabídky Bloky vyberte blok Klonuj (ten s vlastností maskovat) dovnitř bloku Když je v chatu příkaz „kopirovat“. Tento blok klonuje (kopíruje) oblast od prvního bloku souřadnic k druhému a kopíruje ji do třetí sady souřadnic.

24. Z nabídky Proměnné přetáhněte blok proměnné Start do první sady souřadnic „z“ v bloku Klonuj.
25. Z nabídky Proměnné přetáhněte blok proměnné Stop do druhé sady souřadnic „do“ v bloku Klonuj.

Teď můžeme náš kód vyzkoušet.

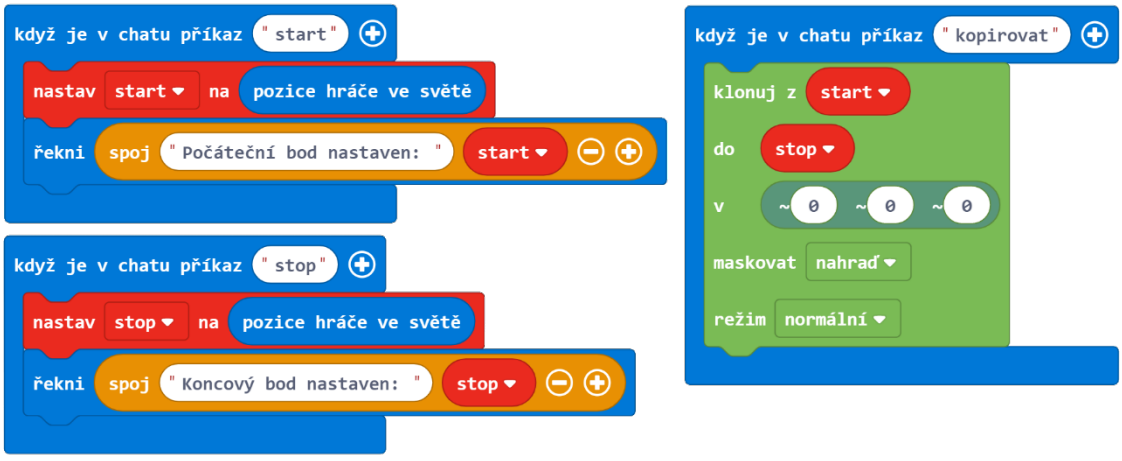
1. Vstupte do Minecraft světa, ve kterém máte strukturu, kterou chcete kopírovat.
2. Přesuňte svou postavu do levého dolního rohu stavby a do chatu napište příkaz „start“.
3. Přesuňte svou postavu do pravého horního rohu stavby a do chatu napište příkaz „stop“.
4. Pak přesuňte svou postavu do otevřeného prostoru ve světě, kam chcete svou stavbu vložit a do chatu zadejte příkaz „kopirovat“.

Zkopírovala se vaše stavba správně?

Poznámky:

- Orientace čehokoliv, co kopírujete, bude vždy stejná. Například, když stojí budova čelem k východu, bude tak natočená i její kopie.
- Pokud v bloku Klonuj ve vlastnosti režim vyberete *přesunout* místo *normální*, stavba z původního světa zmizí a vše mezi dvěma sadami souřadnic se na cílové místo přesune.

Zde je kompletní kód:



JavaScript:

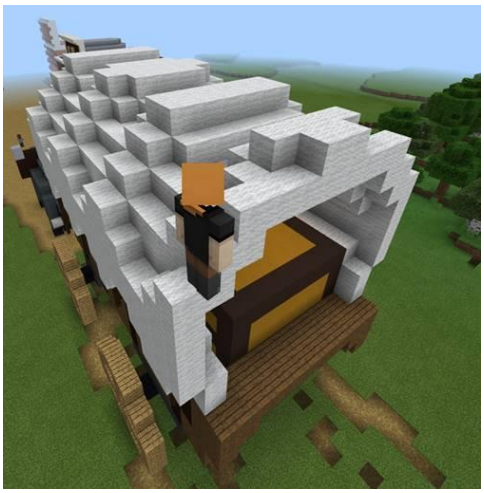
```
let start: Position = null
let stop: Position = null
player.onChat("start", function () {
```

```
    start = player.position()
    player.say("Počáteční bod nastaven: " + start)
})
player.onChat("kopirovat", function () {
    blocks.clone(
    start,
    stop,
    pos(0, 0, 0),
    CloneMask.Replace,
    CloneMode.Normal
    )
})
player.onChat("stop", function () {
    stop = player.position()
    player.say("Koncový bod nastaven: " + stop)
})
```

Sdílený program: https://makecode.com/_dHhhCoW85JAJ

Kopírování vozu ze světa Oregon Trail

Nastavte počáteční blok vlevo dole.



Nastavte koncový blok vpravo nahoře.

Zkopírujte vůz na jiné místo!

Samostatný projekt: Měňte krajinu

Ve svém nezávislém projektu vytvořte jeden nebo více příkazů, které budou nějakým způsobem upravovat krajinu.

Zde jsou některé nápady:

- Vytvořte „instantní plavecký bazén“ vyplněný vodou.
- Vytvořte způsob, jak okamžitě vykopat tunel skrz horu nebo části Netheru.
- Vytvořte způsob, jak doslova hýbat horami.
- Vylepšete aktivitu Stěhovací firma Minecraft, abyste mohli zadat hodnotu koncové souřadnice, místo abyste tam museli sami stát.
- Experimentujte s nastaveními Maska a Režim na bloku Klonuj.

Váš projekt by měl dělat následující:

- Nějak používat souřadnice.
- Nějak upravovat terén.
- Být kreativní a jedinečný.

Příklad:

Tento projekt umožňuje vytvořit sekci cesty s pouliční lampou.

A duplikováním této sekce několikrát za sebou můžete vytvořit cestu jakékoli délky.

Minecraft deník

Zapište si do deníku:

- Proč jste se rozhodli pro tento nápad?
- Jak jste se rozhodli změnit terén?
- Co dělá váš program?
- Popište, jak váš program upravuje terén.
- Zahrňte minimálně jeden snímek výsledku vašeho programu.
- Sdílejte projekt a sem vložte jeho URL odkaz.

POZNÁMKA: Pokud jste se rozhodli vylepšit některou z aktivit v této lekci, promluvte si o tomto novém kódu, který jste napsali. A vysvětlete si toto vylepšení.

Hodnocení

	1	2	3	4
Deník	V deníku chybí odpovědi na 4 nebo více otázek.	V deníku chybí 2 nebo 3 odpovědi.	V deníku chybí 1 odpověď.	Deník obsahuje všechny odpovědi.
Projekt	Projektu chybí všechny žádané prvky: • Použití souřadnic • Úprava krajiny • Kreativní/originální	Projektu chybí 2 ze 3 žádaných prvků: • Použití souřadnic • Úprava krajiny • Kreativní/originálníl	Projektu chybí 1 ze 3 žádaných prvků: • Použití souřadnic • Úprava krajiny • Kreativní/originální	Projekt upravuje krajinu originálním a kreativním způsobem, účinně a efektivně využívá souřadnice.

Proměnné

V této lekci budeme zkoumat proměnné, které jsou důležité pro ukládání informací a pro větší flexibilitu a přizpůsobivost programu. Budeme stavět na tom, co jsme se již dozvěděli o událostech a pomocí proměnné si budeme v průběhu události ukládat informace. Nakonec dostanete úkol, kde si napíšete váš vlastní program s použitím proměnných.

Co je to proměnná?

Počítačové programy zpracovávají informace. Některé informace, které jsou uloženy a používány v počítačovém programu, mají hodnotu, která je konstantní, takže se v průběhu programu nemění.

Příkladem konstanty v matematice je "Pí", protože "Pí" má hodnotu, která se nikdy nemění.

Některé informace mají hodnoty, které se mění během chodu programu. Programátoři vytvářejí proměnné pro udržení hodnoty informací, které se mohou změnit. V počítačových hrách můžeme vytvořit proměnnou pro udržení aktuálního skóre hráče, jelikož se tato hodnota v průběhu hry mění.

Proměnné můžete považovat za kontejnery, které mají různé hodnoty.

Požádejte žáky, aby přemýšleli o informacích v jejich každodenním životě, které bychom mohli považovat za konstanty a proměnné.

- Které informace mají hodnoty, které se v průběhu dne nemění (konstanty)?
- Které informace mají hodnoty, které se mění během dne (proměnné)?
- Konstanty a proměnné mohou být čísla, text nebo logické hodnoty.

Příklady: Během jednoho školního dne

- Konstanty: den v týdnu, rok, jméno studenta, adresa školy.
- Proměnné: teplota, počasí, aktuální čas, aktuální třída, poloha (stojí nebo sedí).

Proměnné uchovávají určitý typ informace. Při prvním použití proměnné je jí přiřazen její typ. Od tohoto okamžiku můžete hodnotu této proměnné měnit pouze na jinou hodnotu stejného typu.

Různé typy proměnných:

Proměnná *číslo* obsahuje číselná data.
Příklady: věk člověka, skóre hráče, rok

Proměnná *řetězec* obsahuje řetězec alfanumerických znaků.
Příklady: jméno člověka, heslo, den v týdnu

Logická (Booleovská) proměnná má pouze dvě možné hodnoty: pravda a nepravda
Příklady: Je už venku světlo? Je konec hry?

V MakeCode pro Minecraft existuje proměnná *Pozice*, která je speciálním druhem proměnné, která obsahuje tři čísla popisující konkrétní umístění v trojrozměrném prostoru. Tato čísla označujeme jako X, Y a Z souřadnice.

Název proměnné	Hodnota
promA	2
promB	4
promC	“ahoj!”
promD	(12, 4, -36)

Proměnné mohou obsahovat různé druhy hodnot – čísla, text, souřadnice atd.

prom A + promB = 6 Řekni promC Teleport to promD	Proměnné mohou být použity místo konkrétních hodnot.
Nastavte promA na (promB + 2) promA = 6	Můžete změnit hodnotu proměnné.

U proměnných, které žáci dříve vymysleli, určete jejich datový typ.

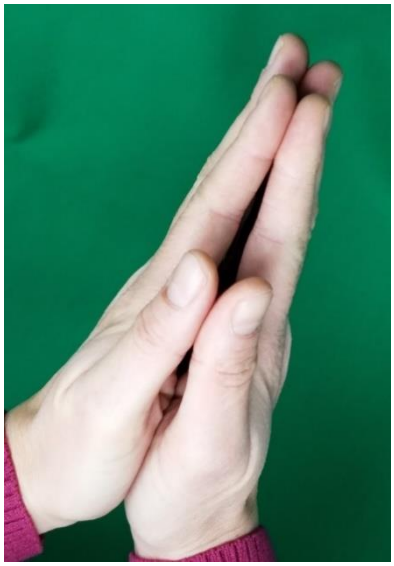
Offline aktivita: Rytmický robot

Jedná se o jednoduchou a zábavnou aktivitu k procvičování používání proměnných pro změnu akce dané funkce.

Tato funkce vytváří jednoduchý rytmus. Skládá se ze 3 pohybů:

- Plácnutí (hráč plácne do stehna, či do stolu)
- Tlesknutí (hráč tleskne rukama)
- Lusknutí (hráč luskne prsty)

Plácnutí



Tlesknutí

Lusknutí

Každému z těchto pohybů je přiřazena proměnná, která řekne hráči (rytmickému robotovi), kolikrát provede daný pohyb:

- Počet plácnutí
- Počet tlesknutí
- Počet lusknutí

Použitím proměnných docílíte toho, že plno různých rytmů může vzniknout pomocí jediné funkce.

Jak hrát:

- Použijte tři různé pohyby v pořadí: Plácnout, Tlesknout a Lusknout.
- Pokaždé, když funkce běží, rytmický robot projde sekvencí Plácnutí, Tlesknutí a Lusknutí 5x.
- Chcete-li určit, kolikrát se bude každý z těchto tří pohybů opakovat během jedné sekvence, nechte studenta vybrat náhodné trojciferné číslo. (Mohli bychom třeba třikrát hodit kostku a získat náhodné trojciferné číslo.)
- První číslice trojciferného čísla bude hodnota Počet plácnutí a určí, kolikrát plácnete do stehen.
- Druhá číslice trojciferného čísla bude hodnota Počet tlesknutí a určuje, kolikrát si tlesknete rukama.
- Třetí číslice trojciferného čísla bude hodnota Počet lusknutí a určuje, kolikrát si lusknete prsty.
- „Spusťte“ svůj program tím, že má student Plácnout, Lusknout a Tlesknout pomocí hodnot proměnných Počet tlesknutí, Počet plácnutí a Počet lusknutí.
- Například: Dejme tomu, že máme náhodné číslo 231. Žáci by plácli dvakrát do stehen, třikrát by si tleskli rukama a pak jednou luskli prsty. Tuto celou sekvenci by opakovali 5x.
- Zvolte nové náhodné číslo, které chcete použít jako hodnoty pro proměnné, a znovu opakujte celý postup.
- Pokračujte ve výběru náhodných čísel, abyste zjistili, jak různé hodnoty pro proměnné ovlivňují výsledky.

Varianty

- Nechte žáky navrhnout různé pohyby, které přidají do sekvence Plácnutí, Tlesknutí, Lusknutí. Mohou pak přidat další proměnnou, jejíž hodnota určí, kolikrát se tento nový pohyb provede.
- Počet opakování celé sekvence Plácnutí, Lusknutí, Tlesknutí může být další proměnná!

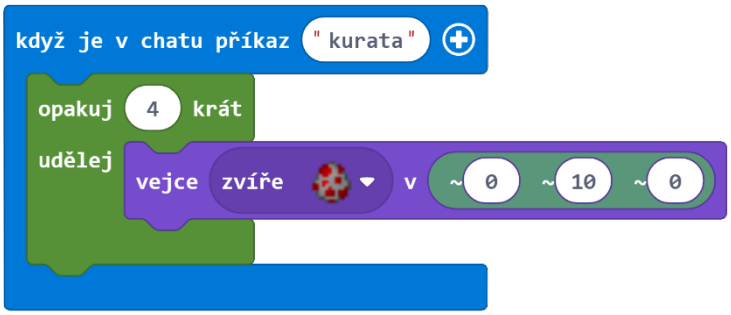
Aktivita: Déšť kuřat

Použijte proměnnou k určení počtu kuřat, které se ve světě Minecraftu objeví. Kuřata se budou rodit ve vzduchu a padat z oblohy. Vytvoříme tak oblíbenou simulaci, kdy nám „pečená kuřata“ budou létat skoro až do pusy.

Postup:

1. Vytvořte nový projekt MakeCode a pojmenujte ho „Kuřecí bouřka“.

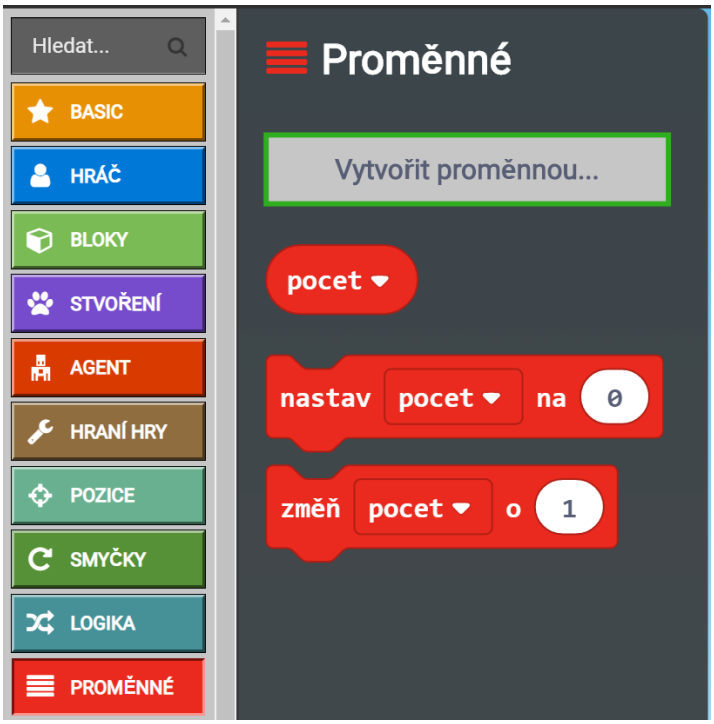
2. Z nabídky na panelu nástrojů přetáhněte blok Když je v chatu příkaz do pracovního prostoru a změňte příkaz na „kurata“.
3. Z nabídky Smyčky vložte příkaz Opakuj do bloku Když je v chatu příkaz "kurata".
4. Z nabídky Stvoření vložte příkaz Vejce dovnitř bloku Opakuj.
5. V bloku Vejce přepište prostřední hodnotu na 10, aby se kuřata objevila 10 bloků nad vaší hlavou.

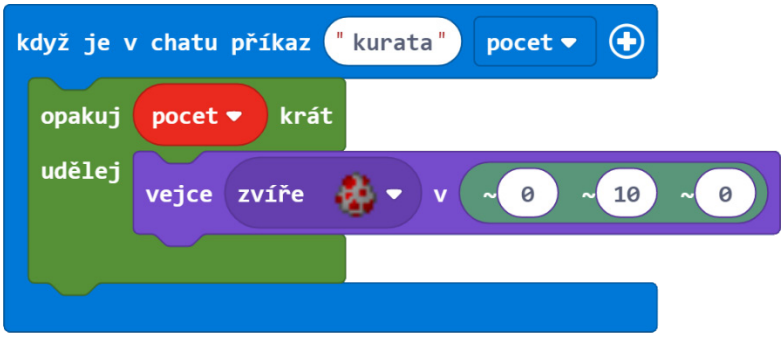


Přejděte do Minecraftu, stiskněte klávesu „T“ pro otevření okna chatu a zadejte „kurata“. Tím spustíte déšť kuřat!

Můžete změnit počet opakování na vyšší, abychom měli více jak 4 kuřata v naší „kuřecí bouřce“. Pojďme o krok dále a umožníme určit hráči, kolik přesně kuřat se má objevit. Přidáme proměnnou bloku Když je v chatu příkaz.

6. Klikněte na znaménko plus (+) vpravo na Když je v chatu příkaz "kurata" bloku. Měla by se objevit rozbalovací nabídka se jménem "num1" v horní části. Num1 je číselná proměnná, kterou můžete použít ve svém kódu. Pro lepší orientaci si proměnnou num1 přejmenujeme.
7. Klikněte na šipku vedle num1, vyberte „Přejmenovat proměnnou…“ a zadejte nový název pocet.
8. Z nabídky Proměnné přetáhněte blok pocet do bloku Opakuj a tímto způsobem nahraďte číslo 4.

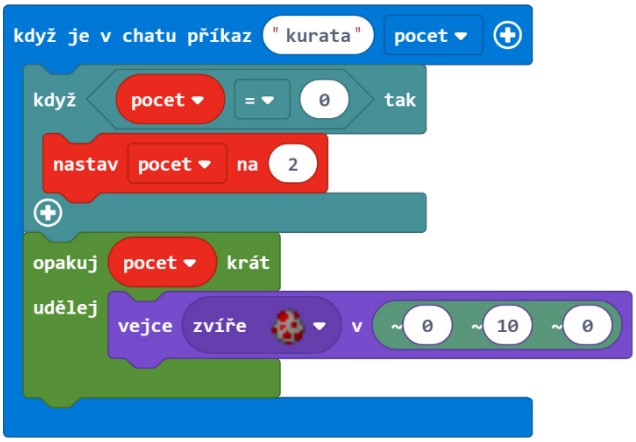




Nyní, když stisknete v Minecraftu opět klávesu „T“ a zadáte „kurata 15“, nastavíte proměnné *pocet* hodnotu na 15. Pokud zadáte „kurata 25“, bude mít proměnná *pocet* hodnotu 25. Číslo, které napíšete po příkazu „kurata“, se nazývá parametr nebo argument. Proměnná je jako kontejner, který drží hodnotu všech parametrů, které jí předáte.

Volitelné rozšíření: Vytvořte výchozí hodnotu kuřat

Nyní jsme vytvořili „kuřecí“ příkaz, který vezme číslo proměnné a vytvoří určité množství kuřat. Je to dobrý trénink na programování, kde si vyzkoušíte, jak zvládnout případy, kde „kuřatům“ není přiřazen jejich počet. Ve výchozím nastavení má pocet hodnotu nula. Pomocí podmíněného příkazu můžete zjistit hodnotu proměnné pocet a v případě, že je nula, nastavit její výchozí hodnotu na 2. Tímto způsobem ošetříte případ, kdy zadáte příkaz „kurata“, ale zapomenete zadat počet kuřat. Dostanete tak alespoň pár kuřat, což je lepší, než nemít žádné a zůstat hlady.



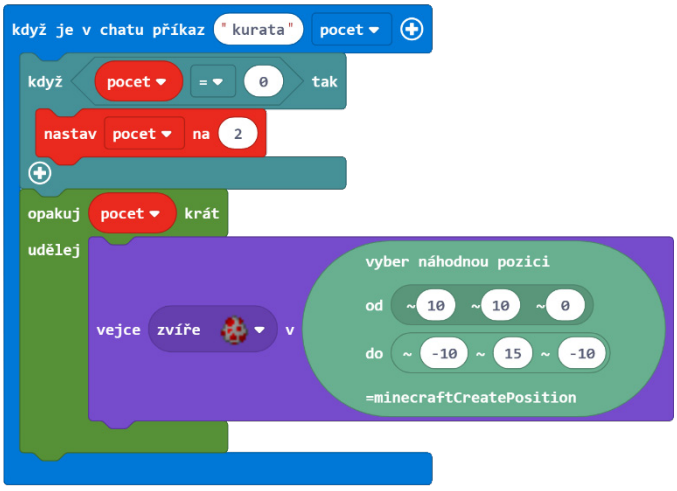
Sdílený program: https://makecode.com/_08WV7RPa53qw

JavaScript:

```
player.onChat("kurata", function (pocet) {
    if (pocet == 0) {
        pocet = 2
    }
    for (let index = 0; index < pocet; index++) {
        mobs.spawn(CHICKEN, pos(0, 10, 0))
    }
})
```

Volitelné rozšíření: Bouře kuřat

Funkci můžete dále vylepšit příkazem Vyber náhodnou pozici, který najdete v nabídce Pozice. Tento blok rozptýlí kuřata náhodně kolem oblasti popsané dvěma souřadnicemi. Můžete také měnit výšku pádu, takže kuřata budou přistávat v různých časech. Nyní je to opravdu bouře kuřat!



Sdílený program: https://makecode.com/_HpD1L6LemCVu

JavaScript:

```
player.onChat("kurata", function (pocet) {
    if (pocet == 0) {
        pocet = 2
    }
    for (let index = 0; index < pocet; index++) {
        mobs.spawn(CHICKEN, randpos(
        pos(10, 10, 0),
        pos(-10, 15, -10)
        ))
    }
})
```

Pro více zábavy s příkazy a proměnnými zkuste tento interaktivní výukový program, který používá příkaz Jump, který vás vystřelí tak vysoko, jak jen chcete! Pozor, pokud se bojíte výšek!

Megaskok:

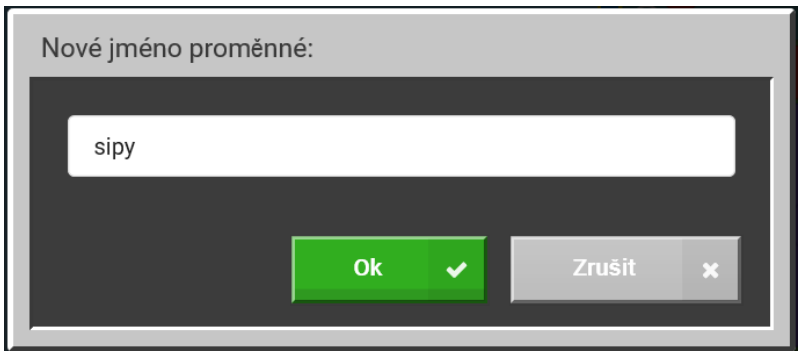
https://minecraft.makecode.com/tutorials/mega-jump

Aktivita: Počítadlo šípů

Pomocí proměnných můžete počítat, kolikrát se ve vašem světě něco děje. Můžete je například použít k udržení skóre, nebo použít podmínku, pomocí které zajistíte akci, jen když se počítadlo dostane na určité číslo. Podívejme se na příklad, který používá počítadlo, které ukládá počet vystřelených šípů.

Kroky:

1. Vytvořte nový projekt MakeCode a pojmenujte ho „Počítadlo šípů“.
2. V nabídce "Proměnné klikněte na tlačítko Vytvořit proměnnou.
3. Nazvěte novou proměnnou „sipy“.



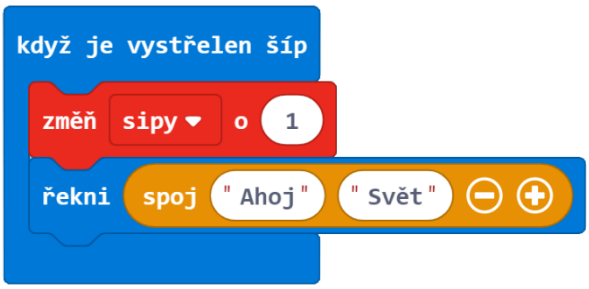
Nyní bude tato nová proměnná („sipy“) dostupná v rozbalovacích nabídkách mnoha bloků, které používají proměnné.

4. Z nabídky Hráč přetáhněte blok Když je vystřelen šíp na pracovní plochu.
5. Z možnosti Proměnné, přetáhněte blok Změň dovnitř bloku Když je vystřelen šíp.
6. V bloku Změň, použijte rozbalovací menu a vyberte název proměnné sipy.

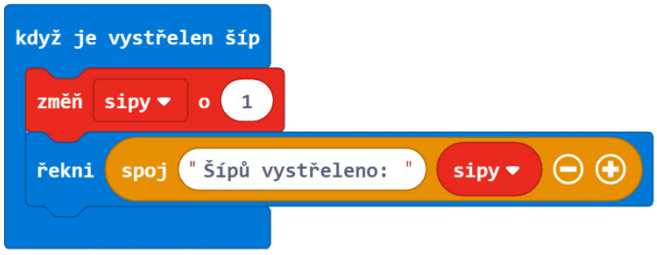
Nyní pokaždé, když vystřelíte šíp, zvýší se počítadlo o 1 (pokud chcete snížit hodnotu proměnné, použijte záporné číslo).

Upravme program tak, abychom viděli aktuální hodnotu proměnné. Blok Řekni vytiskne Text do okna chatu v horní části obrazovky hry Minecraft.

7. Z nabídky Hráč přetáhněte blok Řekni pod blok Změň sipy
   Číselnou hodnotu budeme chtít zobrazit s nějakým textem, aby se nám neobjevovala pouze nic neříkající čísla.
8. Klikněte na nabídku Rozšířené, abyste zobrazili další nabídky včetně nabídky Text.
9. Z nabídky Text, přetáhněte blok Spoj do bloku Řekni a nahraďte text ":)“.



10. V prvním okně bloku Spoj napište „Šípů vystřeleno: “.
11. Z nabídky Proměnné přetáhněte proměnnou „sipy“ do druhého okna bloku Spoj místo řetězce "Svět". Pokaždé, když vystřelíte šíp, objeví se v horní části obrazovky text „Šípů vystřeleno: “ a za ním aktuální hodnota proměnné sipy.



Jděte do Minecraftu a vyzkoušejte to! Pomocí následujících příkazů v okně chatu si můžete vzít luk a šípy:

- /give @s arrow 64
- /give @s bow

Nebo v Tvořivém módu otevřete inventář a luk i šípy vyhledejte.

Sdílený program: https://makecode.com/_1DPAgv2iX9s4

Aktivita: Pád z výšky

Minecraft svět je plný vysokých míst. Pojďme použít zachytávač události, který při pádu z výšky spočítá délku vašeho letu. Pozor, v módu Přežití vás při pádu z velké výšky čeká smrt.

Tento projekt budeme vytvářet v módu Přežití, kde se postavíme na okraj velkého útesu. Můžete ale klidně přepnout na režim Tvořivý a vyletět do libovolné výšky.

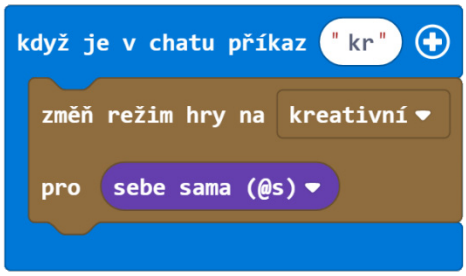
Vytvoříme proměnnou, která bude počítat výšku, z které jsme spadli. Poté, co se znovu oživíme, napíše na obrazovku tuto vzdálenost v metrech (blocích).

Postup:

1. Vytvořte nový projekt MakeCode a nazvěte ho „Pád“.
2. Z nabídky Hráč přetáhněte 3 bloky Když je v chatu příkaz do pracovního prostoru.
3. Přejmenujte tyto příkazy na „kr“ (kreativní), „pr“ (přežití), a „pm“ (posmrtný).
4. Z nabídky Hráč přetáhněte 2 bloky zachytávající události: Když hráč chodí a Když hráč zemře.
5. V rozbalovací nabídce změňte Když hráč chodí na Když hráč padá.

První věc, kterou uděláme, je použití příkazu Když je v chatu příkaz "kr" pro změnu herního režimu na režim Kreativní (Tvořivý). V tomto režimu můžeme létat.

6. Z nabídky Hraní hry přetáhněte blok Změň režim hry na a vložte ho do bloku Když je v chatu příkaz "kr".
7. V bloku Změň režim hry na otevřete rozbalovací nabídku, vyberte mód „kreativní“ a v části pro (koho se daná změna týka) vyberte „sebe sama @s“.



Nyní pro přechod do režimu Kreativní stačí, abyste ve hře napsali do chatu příkaz „kr“. Dvojitým kliknutím na mezerník aktivujete režim létání a budete schopni vyletět tak vysoko, jak jen chcete.

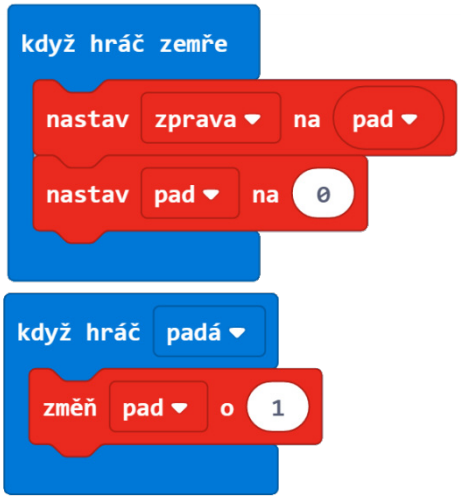
Stejný postup zopakujeme i pro přepnutí do režimu Přežití.

8. Z nabídky Hraní hry přetáhněte blok Změň režim hry na a vložte ho do bloku Když je v chatu příkaz "pr".
9. V bloku Změň režim hry na a opět vyberte selektor „sebe sama @s“.

10. V nabídce Proměnné klikněte na tlačítko „Vytvořit proměnnou“, vytvořte novou proměnnou s názvem „pad“.

Během pádu hráče se tato proměnná začne zvyšovat a bude počítat počet bloků, které hráč padal. Vytvoříme si ještě další proměnnou pro uložení konečného počtu bloků.

11. V nabídce Proměnné klikněte na tlačítko „Vytvořit proměnnou“ a vytvořte novou proměnnou s názvem „zprava“.
12. Z Proměnné přetáhněte blok Změň do bloku Když hráč padá.
13. V bloku Změň rozbalte nabídku a vyberte proměnnou pad.
14. Z Proměnné přetáhněte blok Nastav do bloku Když hráč zemře.
15. V bloku Nastav rozbalte nabídku a vyberte proměnnou zprava.
16. Z Proměnné přetáhněte blok pad do druhého slotu bloku Nastav místo hodnoty 0. Tím si nastavíme hodnotu proměnné pad na 0, aby nám při opakovaném spuštění počítadla pádu počítal vždy od nuly.
17. Z Proměnné přetáhněte další blok Nastav do bloku Když hráč zemře a rozbalte nabídku a vyberte proměnnou pad.



Nakonec musíme vypsat oznámení o výšce pádu na obrazovku. Nezapomeňte, že proměnné mají specifický datový typ a blok Řekni očekává řetězec, nikoliv číslo. Nemůžeme tak do bloku Řekni vložit pouze proměnnou zprava. Budeme muset její hodnotu nejprve převést na řetězec.

18. Z nabídky Hráč přetáhněte blok Řekni do bloku Když je v chatu příkaz "pm".
19. Klikněte na nabídku Rozšířené a zobrazte kategorii Text.
20. Z nabídky Text přetáhněte blok Spoj do bloku Řekni a nahraďte jím slovo "Hi!"

Spoj blok spojí dohromady dvě proměnné a jako výsledek vrátí řetězec. Takže, i když je proměnná zprava číslo, příkaz Spoj ji převede na řetězec.

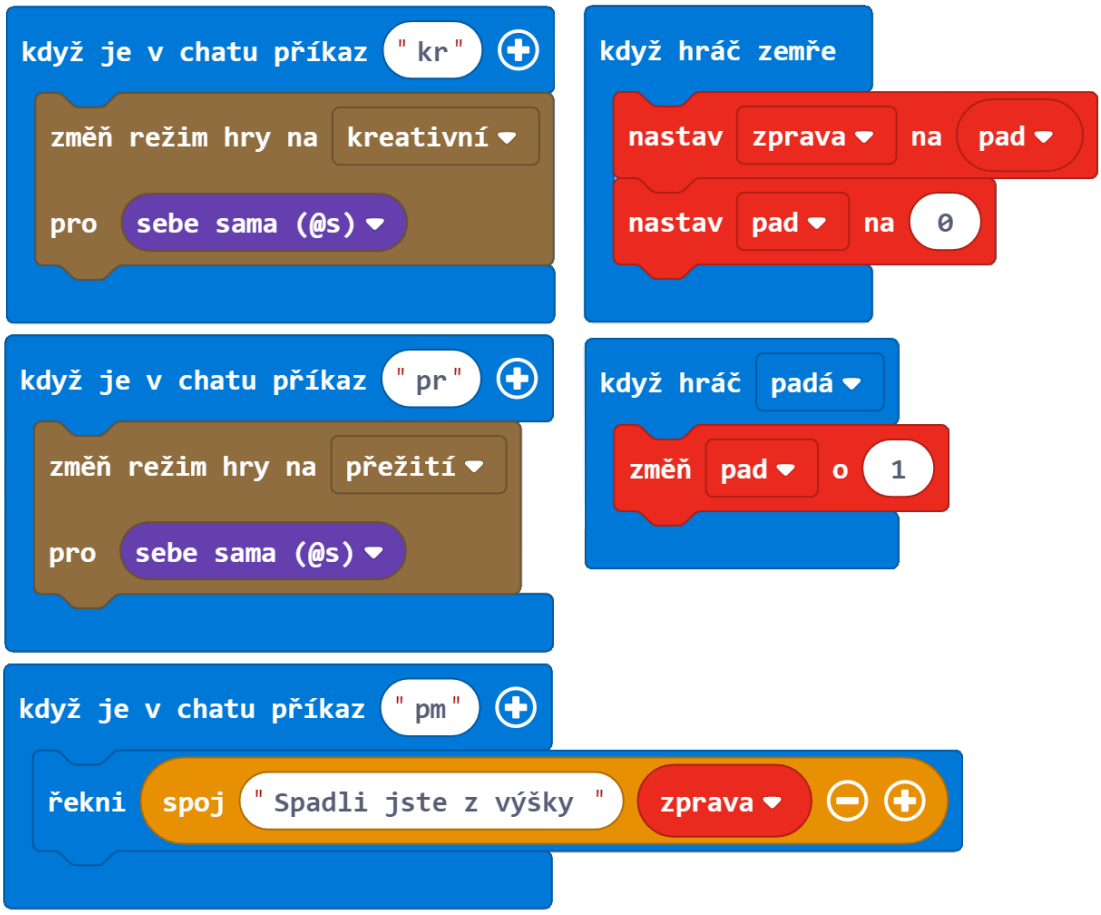
21. Do bloku Spoj zapište místo "Ahoj" "Spadli jste z výšky “.
22. Z nabídky Proměnné přetáhněte proměnnou zprava do druhého slotu bloku Spoj místo slova "Svět".

Toto řešení již bude fungovat, ale výpis bude vypadat takto "Spadli jste z výšky 43". Ale čeho? Měli bychom přidat nějaké jednotky. Abychom toto vylepšení mohli udělat, přepneme se do jazyka JavaScript. (Není nutné přepínat do Javascriptu, abyste přidali řetězec za proměnnou. V bloku Spoj stačí kliknout na znaménko "+". Nicméně tento příklad může být prvním seznámením s textovým programovacím prostředím.).

23. Kliknutím na přepínací tlačítko JavaScript v horní části obrazovky přepnete prostředí MakeCode do editoru JavaScript.
24. Vyhledejte tento řádek kódu: player.say("Spadli jste z výšky " + zprava).
25. Přidejte řetězec na konec tohoto řádku, aby to vypadalo takto: player.say("Spadli jste z výšky " + zprava + " bloků. ")
26. Klepnutím na tlačítko Bloky v horní části obrazovky přepnete zpět do blokového editoru MakeCode.

A je to! Nyní v okně chatu zadejte příkaz „kr“, tím přejdete do módu Kreativní, dvakrát klepněte na mezerník a poté držte mezerník a leťte vzhůru, dokud se nedostanete na oblohu. Poté do okna chatu napište „pr“, přepnete se zpět do módu Přežití a začnete padat. Jakmile zemřete, klikněte na tlačítko Znovuzrodit. Do okna chatu zadejte „pm“. Zobrazí se zpráva o výšce vašeho pádu! Celý program lze zkusit i bez nutnosti zemření. Až vyletíte do nejvyššího bodu, dvakrát opět stiskněte mezerník a začnete padat. V kreativním režimu nemůžete zemřít.

Kompletní program:



JavaScript:

```
let zprava = 0
let pad = 0
player.onChat("kr", function () {
    gameplay.setGameMode(
    CREATIVE,
    mobs.target(LOCAL_PLAYER)
    )
})
player.onChat("pm", function () {
    player.say("Spadli jste z výšky " + zprava)
})
player.onDied(function () {
    zprava = pad
    pad = 0
})
player.onTravelled(FALL, function () {
    pad += 1
})
player.onChat("pr", function () {
    gameplay.setGameMode(
    SURVIVAL,
    mobs.target(LOCAL_PLAYER)
    )
})
```

Sdílený program v ENG: https://makecode.com/_MoyCau9H9VvP

Samostatný projekt: Používání proměnných

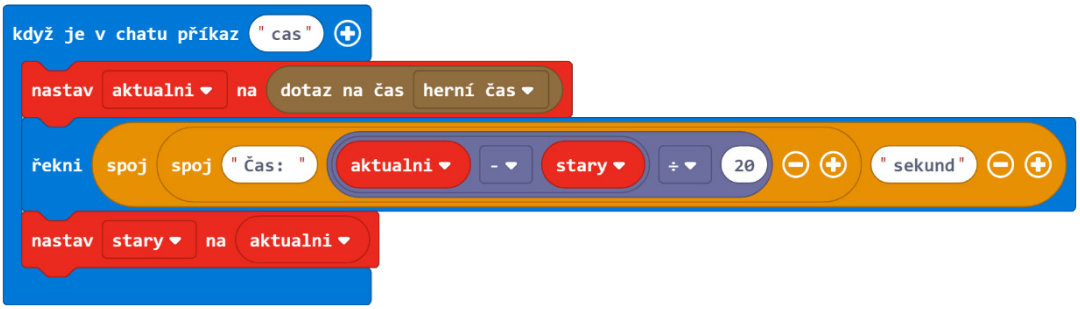
V této lekci jsme se zabývali tím, jak vytvořit novou proměnnou a jak změnit hodnotu již vytvořené proměnné. Naučili jsme se předávat hodnotu proměnné do Když je v chatu příkaz bloku jako parametr a také jsme se podívali na způsob, jak pomocí proměnné sledovat události, které ve hře nastanou.

Nyní vytvořte originální projekt v MakeCode, který využívá více proměnných alespoň dvou různých typů a sleduje situace v Minecraftu. Vytvořte funkci, která bude pro svůj běh vyžadovat zadání parametru (může to být ve stejném projektu nebo jiném projektu).

Příklad:

Alpský závod

Střídejte se ve skákání na vrchol hory. Sledujte uplynulý čas, stejně jako počet skoků, které každá osoba udělala od začátku až do konce. Zde je příklad kódu, který můžete použít k měření času.



Tento kód využívá dvou proměnných, z nichž jedna drží aktuální čas hry a druhá drží starou hodnotu času. Když v okně chatu zadáte „cas“, zobrazí se počet sekund, které uplynuly od posledního zvolání příkazu „cas“.

Pro měření času závodu stačí od sebe obě proměnné odečíst a vydělit 20, protože v Minecraftu jedna sekunda obsahuje 20 herních ticků.

Minecraft deník

Zapište si do deníku:

- Jaký druh informace jste se rozhodl sledovat? A jak?
- Na jaké problémy jste narazili? Jak jste je vyřešili?
- Jaké jste použili proměnné ve svém projektu? Jaké byly jejich datové typy?
- Co bylo nové, co jste se naučili v tomto projektu? Popište, jak jste to řešili.
- Vložte alespoň jeden snímek obrazovky vašeho projektu.
- Sdílejte svůj projekt na webu a vložte zde adresu URL.

Hodnocení

	1	2	3	4
Diář	V deníku chybí odpovědi na 4 nebo více otázek.	V deníku chybí 2 nebo 3 odpovědi.	V deníku chybí 1 odpověď.	Deník obsahuje všechny odpovědi.
Proměnné	Projekt neobsahuje proměnné.	Nejméně jedna proměnná je implementována smysluplným způsobem nebo všechny proměnné jsou stejného typu.	Nejméně 2 různé proměnné jsou implementovány smysluplným způsobem. Proměnné jsou různých typů (text, číslo, boolean, poloha)	Nejméně 3 různé proměnné jsou implementovány smysluplným způsobem. Proměnné jsou různých typů (text, číslo, boolean, poloha)
Příkaz/ Parametr	Projekt nepoužívá žádný příkaz.	Projekt používá příkazy z chatu, ale neimplementuje parametry.	Projekt používá příkazy z chatu s jedním nebo více parametry, které nejsou v kódu použity.	Projekt zahrnuje příkaz z chatu, který používá jeden nebo více parametrů v přiloženém kódu.

Opakování (cyklus, iterace)

V této lekci prozkoumáme možnosti opakování. Můžete opakovat sadu instrukcí v programu k docílení určitého efektu, nebo můžete použít opakování k provedení stejného úkolu v menším počtu kroků. Seznámíme se s Agentem, vaším osobním robotem, který vám pomůže plnit běžné úkoly jako budování farmy, stavba domů nebo těžba surovin. V této lekci naučíme Agenta originální taneční pohyby, aby se mohl předvést třeba na tanečním parketě a ohromit protihráče v Minecraftu. V závěru si sami naprogramujete Agenta a naučíte ho splnit vámi vybranou užitečnou úlohu.

Co je to cyklus?

Učitelé informatiky často říkají: „Cyklus je opakování. Cyklus je opakování. Cyklus je opakování.“ Chápete to. Chcete-li vytvořit cyklus, opakujte.

A opakování je něco, co počítače umí velmi dobře a velmi rychle.

Poznámka: Opakovat versus zopakovat

V programování budeme používat slovo Opakovat (Iterace) namísto slova Zopakovat. (Reiterace). Jedná se o ustálený termín, který se v informatice používá.

Jaké jsou některé příklady opakování z reálného života?

Příklad: Chůze. Nemyslíme na to, ale naše těla opakují stejnou sekvenci akcí, kdykoli chodíme. Základní viditelná akce chůze je složena z opakování akcí „levou nohou dopředu, pravou nohou dopředu“.

Jaké jsou další příklady cyklů ve vašem každodenním životě? Jaké úkoly a akce se skládají z opakovaných kroků? Jaké jsou tyto kroky?

Architekti používají cykly (iteraci) v návrzích staveb.

Psi se rádi honí opakovaně za klackem.

V přírodě existuje mnoho opakovaných (iteračních) vzorků.

Offline: Každodenní úkoly

Požádejte žáky, aby napsali do sešitu nějakou činnost a na samostatný list popsali opakované kroky, kterou tato činnost tvoří.

Nechte je, aby si vyměnili seznam opakovaných kroků se spolužákem.

Každý žák se bude snažit přijít na to, jakou běžnou činnost jeho spolužák popsal pomocí opakování kroků.

Poznámka učitele:

V závislosti na věku žáků a čase můžete:

1. Určit každému žákovi každodenní činnost k popsání kroků opakování. Tím se vyhnete tomu, že většina žáků bude mít podobnou úlohu.

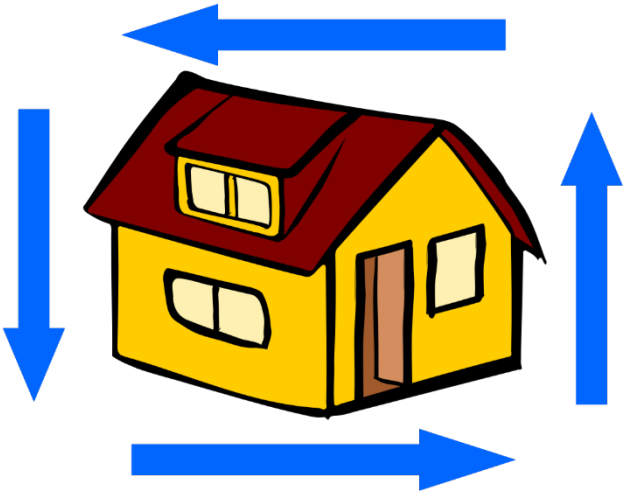
2. Přečtěte některé postupy před celou třídou. Můžete vybrat jednoho nebo všechny žáky, kteří budou postupovat podle těchto kroků.

Některé příklady úkolů:

- Kývnutí hlavou ano
- Kývnutí hlavou ne
- Mávaní někomu
- Chození po schodech nahoru/dolů
- Vykonávat určité cvičení (skákat panáka)
- Česat si vlasy/Čistit si zuby
- Jíst polévku z misky pomocí lžičky
- Stříhání papíru nůžkami

Offline aktivita: Chůze okolo domu

Tato aktivita je pro dvojice. Jeden bude zapisovat a druhý bude diktovat pokyny, jak obejít dům v Minecraftu a vrátit se na začátek. Budete potřebovat židli a sešit pro zápis pokynů. Umístěte židli do středu místnosti (stačí volné místo tak, aby se dalo okolo židle projít). Židle bude představovat dům v Minecraftu. První z obou žáků si stoupne do jednoho z rohů domu.



Úkolem druhého je zapsat pokyny od spolužáka. K pohybu však smí využívat pouze těchto dvou příkazů:

```
Vpřed
Zahni vlevo/vpravo
```

Řada instrukcí, podle kterých počítač vykonává své operace, se nazývá algoritmus. V jednom kroku může počítač zpracovat pouze jednu instrukci. Algoritmus musí mít tedy v každém kroku přesně určeno, co se má vykonat.

Váš pseudokód bude pravděpodobně vypadat takto:

```
vpřed
zahni vlevo
vpřed
zahni vlevo
vpřed
zahni vlevo
vpřed
zahni vlevo
```

Nechte svého spolužáka jít přesně podle algoritmu. Tím ověřte, zda funguje.

Všimněte si, že se jedná o osm řádků kódu a některé řádky se opakují. Následující dva řádky kódu se opakují čtyřikrát:

```
vpřed
zahni vlevo
```

Programátoři rádi používají zkratky. Méně řádků kódu zabírá méně místa v paměti počítače a nakonec, čím je váš program kratší, tím je snazší odhalit případné chyby. Kdykoli máte kód, který se opakuje, můžete použít cyklus k zjednodušení kódu.

Kód s použitím cyklu by mohl vypadat takto:

```
Opakujte 4krát:
    vpřed
    zahni vlevo
```

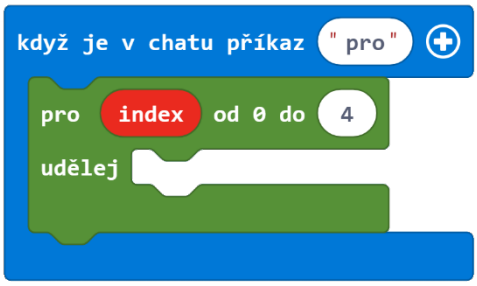
Postupujte podle tohoto opraveného algoritmu a opět otestujte jeho funkčnost.

Právě jste přepsali osm řádků kódu na tři řádky s cyklem. Příkaz 'opakovat' vytvoří smyčku (cyklus). Kód v rámci smyčky se opakuje tolikrát, dokud není splněna podmínka. Podmínkou v tomto algoritmu je přesný počet opakování (4). Jakmile je tato podmínka splněna, program smyčku opustí.

Druhy cyklů v MakeCode

Počítače používají různé cykly k opakování sady instrukcí. V této části si projdeme, které cykly můžete v MakeCode najít. Zamyslete se, které typy činností by byly vhodné pro každý typ cyklu.

Opakujte n krát



Pro index 0 do n

Zatímco cyklus Opakuj se opakuje n krát, cyklus Pro se opakuje n+1 krát (počítá se od 0). Pokud máte ve vašem domě 32 schodů, pak jít nahoru nebo dolů znamená učinit 1 krok 32krát. Další příklady činností, které mají pevný počet opakování: zavázání si tkaniček, otáčení stránek v obrázkové knize, běžkování, slalom a další.

Podobně jako u bloku Opakuj také blok Pro provádí přesný počet opakování. Používá ale navíc proměnnou s názvem index (její název lze změnit), kterou můžete použít uvnitř smyčky, pokud potřebujete pracovat s informací, v kterém kroku opakování se právě nacházíte.

Zajímavost: V jazyce JavaScript je cyklus Opakuj ve skutečnosti stejný jako cyklus Pro! V blokovém prostředí je pro zjednodušení proměnná u cyklu Opakuj skryta. Můžete si to sami prohlédnout po přepnutí se do Javascriptu v horní části okna.

```
player.onChat("pro", function () {
    for (let index = 0; index <= 4; index++) {

    }
})
player.onChat("opakuj", function () {
    for (let index = 0; index < 4; index++) {

    }
})
```

Dokud <pravda>

Cyklus s podmínkou („dokud“) běží tak dlouho, dokud je podmínka splněna (pravdivá). Ve výchozím nastavení je vždy pravdivá. Tímto způsobem můžete vytvořit nekonečný cyklus. Obvykle však nahrazujeme blok <pravda> jinou podmínkou, která nabývá jak hodnoty pravda, tak nepravda.

Například: Při podmínce <ruce jsou špinavé> bychom se myli tak dlouho, dokud podmínka <ruce jsou špinavé> je nepravda (ruce jsou čisté).

Některé příklady aktivit, v kterých bychom mohli použít cyklus „dokud“:

- dokud <zbývá polévka>, jezte.
- dokud <energie větší než 0>, skákejte panáky.
- dokud <ještě mě neviděla>, zuřivě mávejte.

V Minecraftu můžete vytvořit například cyklus dokud, který se opakuje tak dlouho, dokud Agent nerozpozná pod sebou rudit. Můžete si představit cyklus dokud jako cyklus Opakuj v kombinaci s podmíněným příkazem když – Když je pode mnou rudit, udělej toto ....

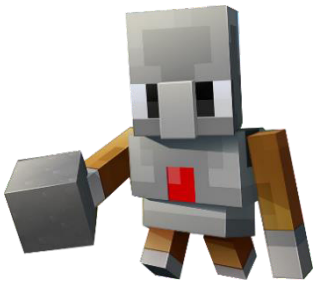
Opakuj stále

Je další ukázka nekonečného cyklu. Příkazy uvnitř tohoto cyklu běží po celou dobu spuštěného programu.

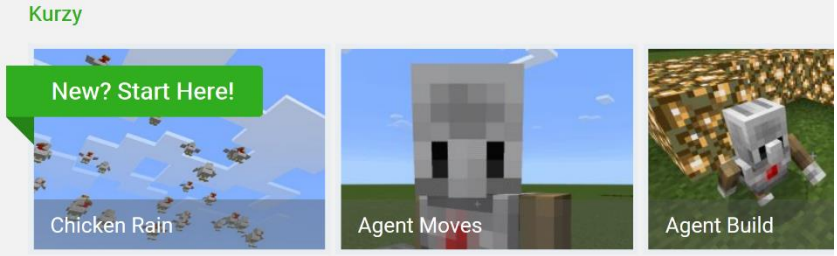
Některé příklady nekonečných cyklů z reálného života: dýchání, činnost srdce.

Aktivita: Seznamte se s postavou Agenta

Minecraft Agent je malý robot, který za vás může plnit příkazy, které napíšete v MakeCode. V této části si ukážeme základy ovládání Agenta.

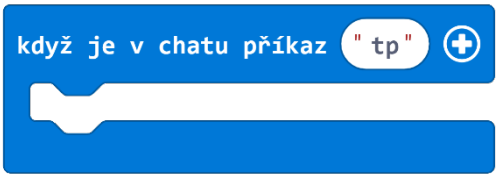


Jinou možností je podívat se na interaktivní návod v kurzech přímo na domovské stránce MakeCode.



Postup:

1. Vytvořte nový projekt v MakeCode s názvem „Ovládání Agenta“.
2. Přejmenujte původní blok Když je v chatu příkaz "tp".



3. Z nabídky Agent přetáhněte blok Agente teleportuj se k hráči a vložte ho do bloku Když je v chatu příkaz.

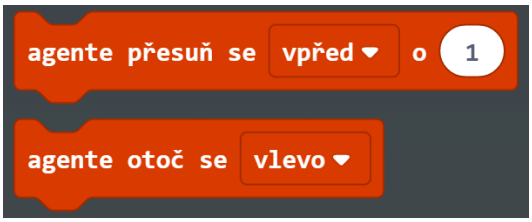
V Minecraftu otevřete pomocí klávesy T okno chatu a zavolejte příkaz „tp“. Agent se teleportuje na vaši pozici. To je nejjednodušší způsob, jak si Agenta přivolat. Obvykle tím začíná většina projektů.

Podívejte se na další základní příkazy pro pohyb Agenta:

Aktivita: Tancuj Agente, tancuj!

V této aktivitě naučíme Agenta jedinečný tanec, který využívá opakování (konkrétně cyklu Pro). V nabídce Agent naleznete následující příkazy:

- Přesuň se (vpřed) o 1
- Otoč se (vlevo)



Pro každý z těchto příkazů můžete z rozbalovací nabídky zvolit jiný směr. Příkazem Agent přesunout se nezmění směr, kterým se Agent dívá. Použijete-li tedy příkaz agente přesuň se vlevo, Agent udělá úkrok vlevo (půjde bokem).

Příkazem otoč se se už Agent skutečně otočí. Pokud například chcete, aby se Agent otočil dokola ve směru hodinových ručiček, použijte čtyřikrát za sebou příkaz agente otoč se vpravo.

V nabídce je celá řada dalších příkazů pro Agenta, které provádějí specifické užitečné akce a umožní Agentovi útočit nebo orat půdu:

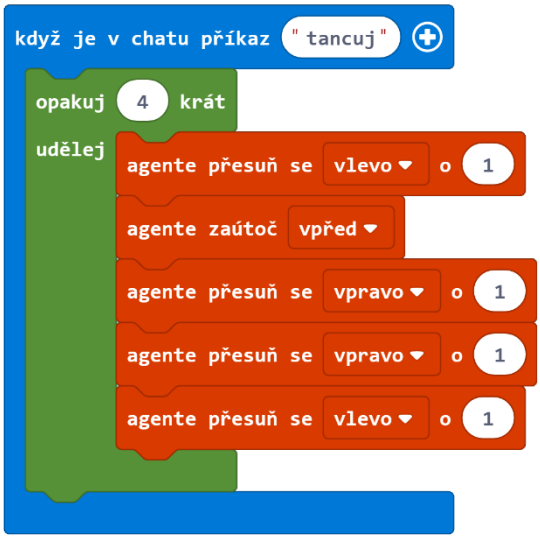
Pro naše taneční účely se nebojte kombinovat některé z těchto příkazů, abyste svůj tanec udělali ještě více zajímavým. Začněme!

Postup:

1. Z nabídky Hráč přetáhněte nový blok Když je v chatu příkaz do pracovního prostoru.
2. Pojmenujte tento příkaz "tancuj".

3. Vytvořte cestu (trajektorii), po které půjde váš agent. Zde je jeden příklad:

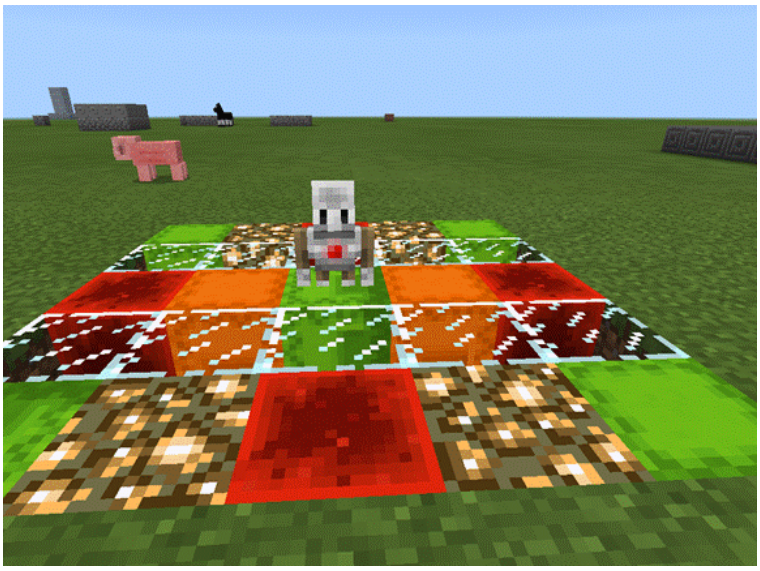
4. Z nabídky Smyčky přetáhněte blok Opakuj do bloku Když je v chatu příkaz a obklopte tím bloky Agenta.



Pomocí bloku Opakuj opakujte několikrát taneční pohyby. Přepněte se do Minecraftu, teleportujte Agenta na vaši pozici a nechte ho zatančit! (Do chatu napište příkaz „tancuj“).

Dejte dohromady všechny své nejlepší taneční pohyby a vytvořte úžasné taneční představení!

Co takhle vytvořit osvětlený taneční parket?



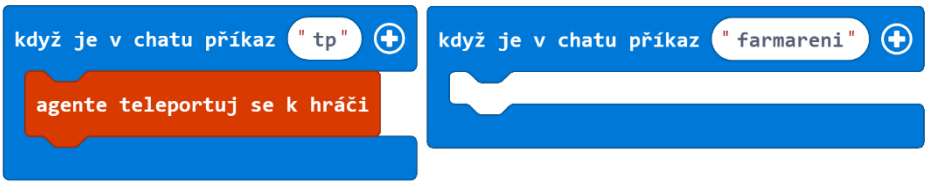
Aktivita: Agent farmář

Ke splnění této úlohy se snažte maximálně využívat cykly. Naučíme Agenta zrýt několik řad půdy, aby zde mohl zasadit plodiny. Při ladění tohoto kódu (zároveň zjistěte, kde jsou problémy!) budeme potřebovat vaši pomoc. Pojďme na to!

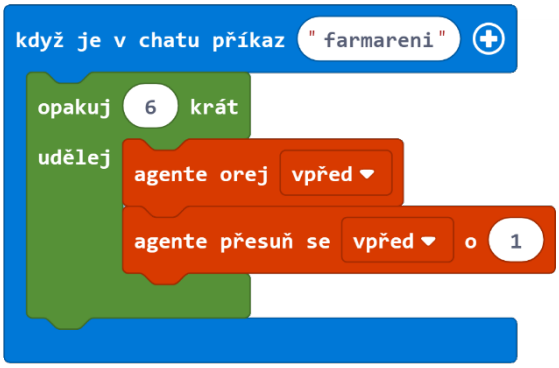
Chtěli bychom vytvořit záhon (pole) upravené půdy o velikosti 2 x 6. Celý proces si můžeme představit jako několik se opakujících smyček, díky kterým agent projde jednu řadu a zryje ji, pak se otočí a znovu opakuje stejné kroky pro druhou řadu.

Postup:

1. V MakeCode vytvoříme nový projekt s názvem „Farmář“.
2. Z nabídky Hráč přetáhněte další blok Když je v chatu příkaz do pracovního prostoru. Tím si připravíme dva bloky Když je v chatu příkaz.
3. První pojmenujte "tp" pro teleportaci Agenta na vaše místo, druhý pojmenujte "farmareni".
4. Z nabídky Agent přetáhněte blok Agente teleportuj se k hráči a vložte ho do bloku Když je v chatu příkaz "tp".



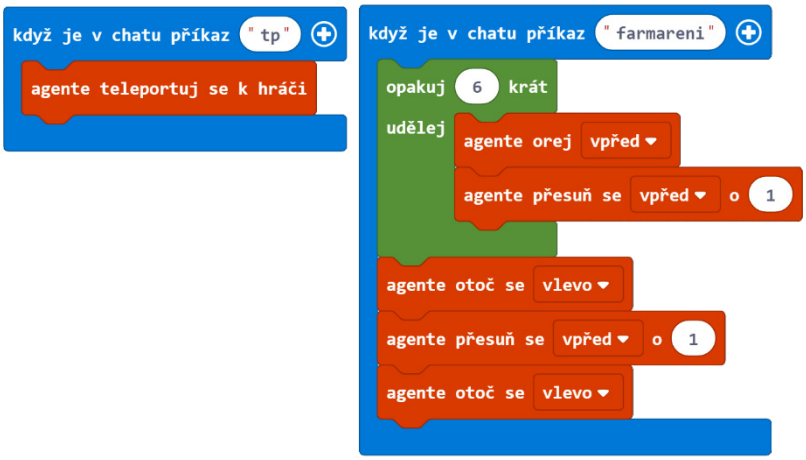
5. Z nabídky Smyčky přetáhněte blok Opakuj do bloku Když je v chatu příkaz "farmareni".
6. Tím určíte délku řady záhonu. V bloku Opakuj změňte číslo 4 na 6.
7. Z nabídky Agent přetáhněte blok Agente orej a blok Agente přesuň se do bloku Opakuj.



Tato smyčka se bude opakovat 6x. Pokaždé Agent upraví půdu přímo před ním (vlastní diamantovou motyčkou) a udělá krok vpřed. Spusťte kód v Minecraftu a uvidíte, co se stane.

Všimněte si, kde Agent skončí po dokončení bloku Opakuj. Chceme, aby se otočil a upravoval další řadu záhonu a tím vytvořil oblast 2 x 6 bloků upravené půdy připravené k výsadbě. Budete muset přidat další blok Opakuj kolem vašeho stávajícího bloku Opakuj a přidat některé příkazy, abyste Agenta otočili předtím, než začne další řadu.

Výsledný kód:

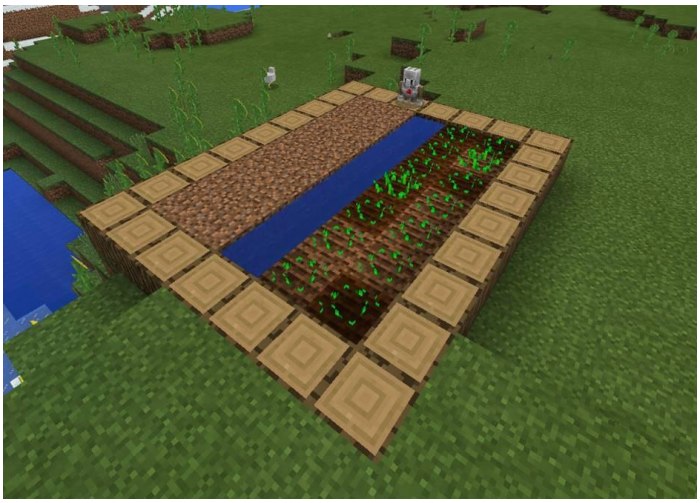


Pokud spustíte tento kód, zjistíte, že nefunguje úplně správně. Agent po otočení není správně nastaven a druhá řada záhonu není zarovnána s tou první. Jak můžete opravit kód, aby fungoval správně? To už necháme na vás, abyste na to přišli sami!

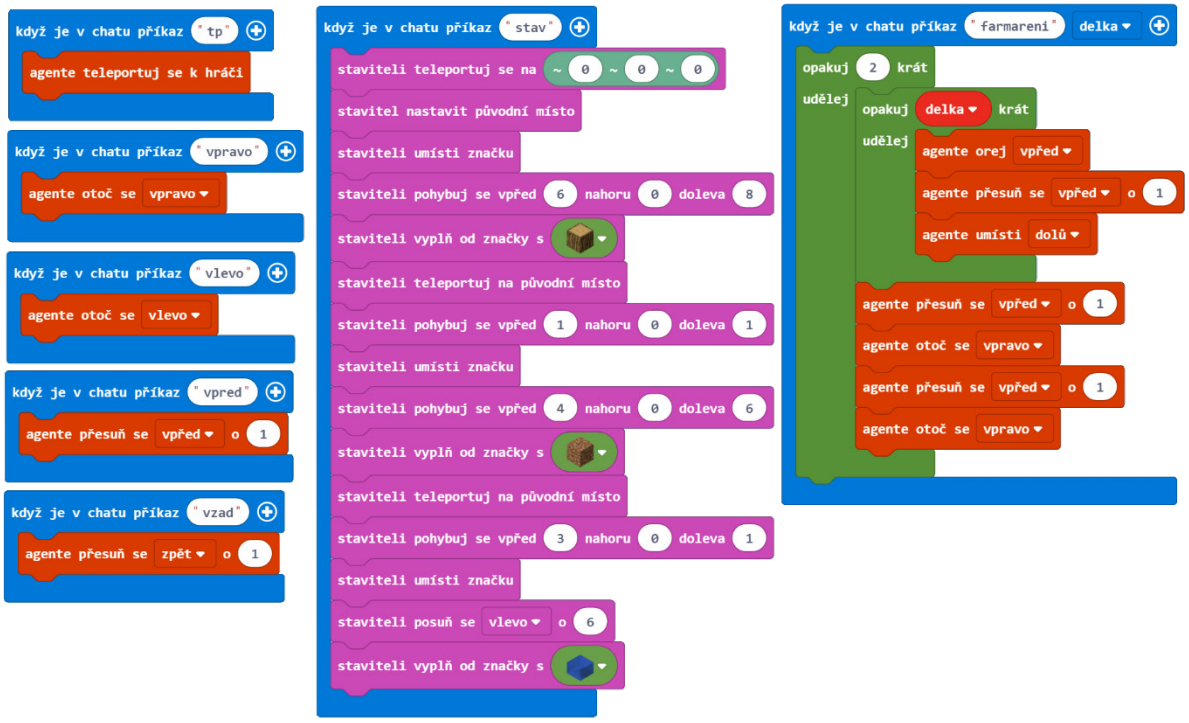
Řešení pro učitele: https://makecode.com/_2z3Hg6W4U1YV

Volitelné rozšíření:

1. Plodiny potřebují k růstu vodu. Naprogramujte Agenta tak, aby mezi každými dvěma řadami zorané hlíny vytvořil řadu vody.
2. Naučte Agenta, aby poté, co upraví půdu, vysadil několik semen. Nezapomeňte Agentovi předat semena do levého horního slotu jeho inventáře.
3. Vytvořte kód, který bude Klonovat farmářské záhony, které jsou ve vesnicích. To už není tak snadné, ale zkuste k tomu využít Stavitele. Jedno možné řešení můžete nalézt níže, ale určitě existují i další. Vyzkoušejte si to!



Kompletní program:



JavaScript:

```
player.onChat("vlevo", function () {
    agent.turn(LEFT_TURN)
})
player.onChat("vpred", function () {
    agent.move(FORWARD, 1)
})
player.onChat("stav", function (delka) {
    for (let index = 0; index < 2; index++) {
        for (let index = 0; index < delka; index++) {
            agent.till(FORWARD)
            agent.move(FORWARD, 1)
            agent.place(DOWN)
        }
        agent.move(FORWARD, 1)
        agent.turn(RIGHT_TURN)
        agent.move(FORWARD, 1)
        agent.turn(RIGHT_TURN)
    }
})
player.onChat("vzad", function () {
    agent.move(BACK, 1)
})
player.onChat("tp", function () {
    agent.teleportToPlayer()
})
player.onChat("farmareni", function () {
    builder.teleportTo(positions.create(0, 0, 0))
    builder.setOrigin()
    builder.mark()
    builder.shift(6, 0, 8)
    builder.fill(LOG_OAK)
    builder.teleportToOrigin()
    builder.shift(1, 0, 1)
    builder.mark()
```

```
    builder.shift(4, 0, 6)
    builder.fill(DIRT)
    builder.teleportToOrigin()
    builder.shift(3, 0, 1)
    builder.mark()
    builder.move(LEFT, 6)
    builder.fill(WATER)
})
player.onChat("vpravo", function () {
    agent.turn(RIGHT_TURN)
})
```

Sdílený program: https://makecode.com/_FTA0EuARH7Ys

Samostatný projekt: Schodiště

V tomto projektu máte příležitost ukázat, co jste se již naučili. V této lekci jste se dozvěděli, jak mohou bloky cyklů snížit množství opakování ve vašem kódu. Nyní je vaším úkolem přijít s projektem, který nějakým způsobem využívá cyklů pro vytvoření schodiště k nalezišti diamantů.

Jedním z nejcennějších zdrojů v Minecraftu je diamant. Diamanty naleznete v režimu Přežití. Dostat se z povrchu až dolů do míst, kde se nachází diamanty, je velice náročný úkol. Naštěstí můžete naprogramovat svého Agenta, aby vám pomohl! V tomto projektu najděte způsob, jak naučit Agenta vykopat tunel do úrovně Y12 nebo Y13, kde jsou k nalezení diamanty. Jakmile se dostanete na tuto úroveň, můžete začít kopat paralelní tunely a hledat diamanty nebo dokonce naprogramovat Agenta tak, aby kopal za vás (vyřešit tuto úlohu se naučíme v kapitole o podmínkách).

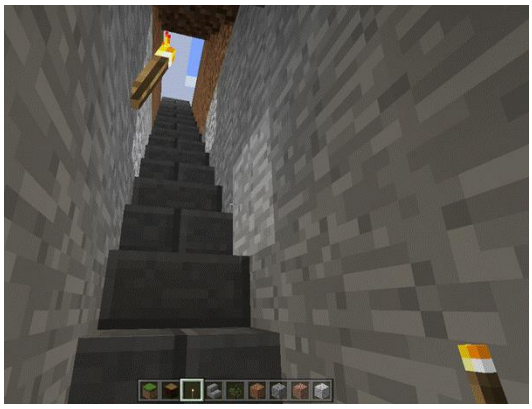
Poznámka: V Minecraftu obecně není moc dobrý nápad kopat rovně dolů, protože můžete narazit na podzemní jeskyně. Může se tak stát, že si nebudete schopni umístit žebřík, abyste vylezli nahoru. Používat schodiště je obvykle lepší způsob. Přesto musíte jít po Agentovi a cestu za ním trochu uklidit. Může se stát, že se prokope do jeskyně, kde nejsou bloky a nebude moct vytvářet schody, nebo za ním zasype cestu žula či písek.

Projekt Schodiště: Možnost #1

Rovné schodiště

Rovné schodiště je nejvíce podobné tomu, na které jsme zvyklí. Výhodou je, že vám umožní rychle stoupat a klesat, pokud máte dostatečně vysoký strop.

Pohled dolů ze schodů.



Pohled nahoru do schodů.

Tento úkol je trochu jednodušší, pokud neumisťujete bloky schodů (speciální blok s menší výškou), po kterých můžete pomocí klávesy W jít bez problémů nahoru i dolů. Na druhou stranu ale budete muset na každý schod vyskočit, a to zvýší vaši spotřebu jídla (v režimu Přežití). Také je nepraktické přivést dolů do dolu koně, pokud nemáte široké, vysoké schodiště postavené z bloků schodů.

Pokud se rozhodnete pokládat schody, berte na vědomí, že Agent ve výchozím nastavení umístí schody směřující dozadu. To není problém, pokud stavíte schodiště směrem nahoru. Ale pokud stavíte schodiště směrem dolů, přemýšlejte o tom, jak byste mohli upravit svůj kód, abyste se ujistili, že schody vypadají správně.

Nezapomeňte také, že když blok Agente umísti při pohybu je nastaven na hodnotu PRAVDA, Agent umístí blok při každém pohybu. Možná budete muset přepínat v bloku Agente umísti při pohybu mezi hodnotami Pravda a Nepravda, abyste nejprve Agenta natočili správným směrem a až poté položil blok.

Nezapomínejte používat cykly, abyste omezili množství opakujícího se kódu!

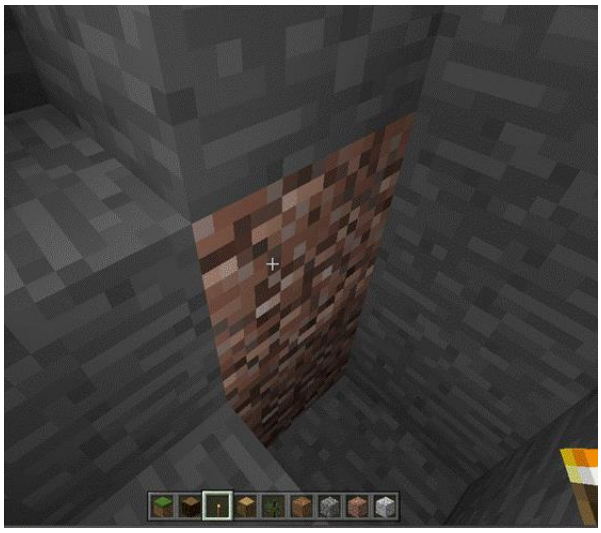
Až budete mít hotové rovné schodiště, přemýšlejte nad tím, jak byste mohli pozměnit kód tak, abyste vytvořili schodiště 3 bloky široké s dostatkem místa pro jízdu na koni.

Projekt Schodiště: Možnost #2

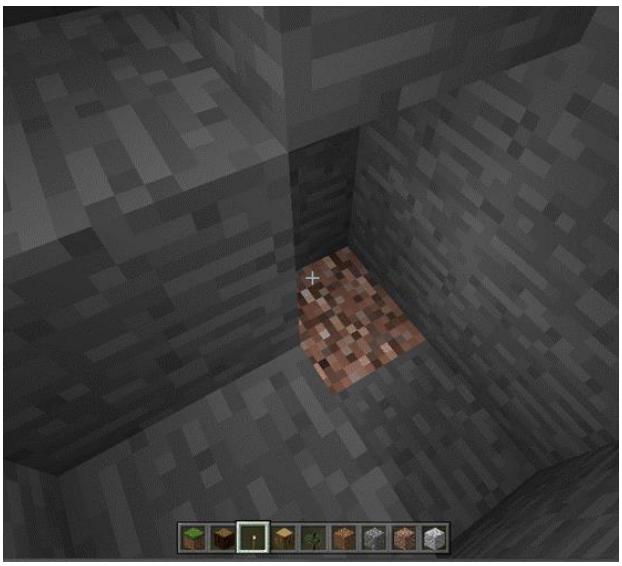
Točité (spirálovité) schodiště

Jiným typem schodiště je točité schodiště. Vymyslete, jak můžete Agenta naprogramovat tak, aby tento typ schodiště vykopal za vás.

Začněte tím, že vykopete 1 blok, pak 2 bloky, pak 3 bloky hluboko.



U každého otočení odstraňte sloupec tří žulových bloků (na dalším obrázku je vidět blok, který v prvním není viditelný, dokud neodstraníte právě tyto tři žulové bloky).

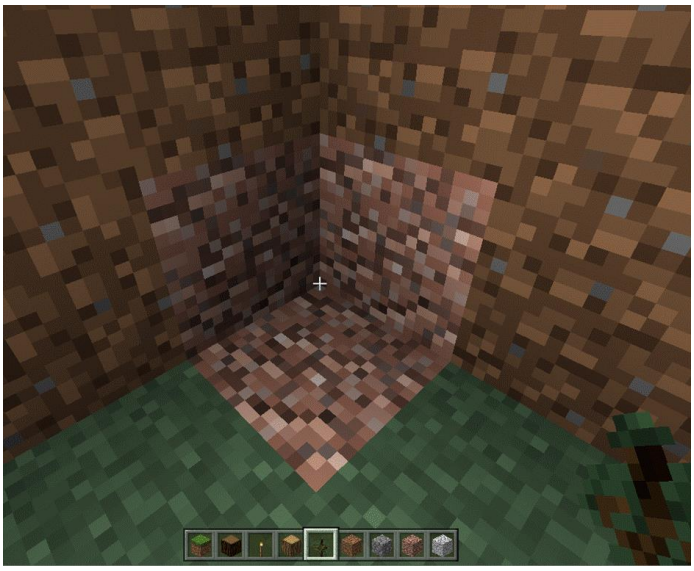


Točité schodiště zahrnuje část kódu, kterou byste použili k vytvoření přímého schodiště širokého jeden blok. Přemýšlejte nad tím, kam se v každém kroku otočit. A stejně jako u všech schodišť, budete pravděpodobně muset následovat Agenta, abyste umístili pochodně a vyčistili prostor po pádu žuly či písku.

Projekt Schodiště: Možnost #3

Diagonální (úhlopříčný) tunel

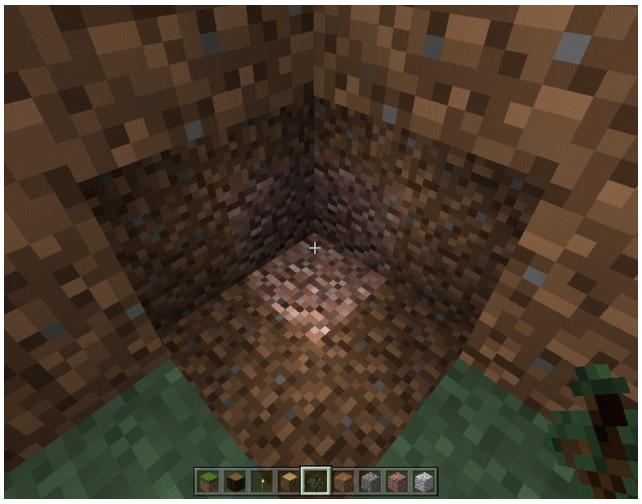
Jeden z typů tunelů (schodišť) v Minecraftu jde šikmo pod povrch. Dokážete napsat kód, díky němuž bude Agent vykopávat tento typ tunelu? Kopání diagonálního tunelu zahrnuje odstranění vrstev bloků tak, jak je znázorněno na obrázcích.



Krok 1: Odstraňte žulové bloky.

Krok 2: Odstraňte kamenné bloky.

Krok 3: Odstraňte prostřední blok.



Krok 4: Opakujte tento postup pro nové tři bloky.

Všimněte si, že Agent nemůže stát diagonálně jako vy, takže budete muset přijít na to, jak přesunout Agenta tak, aby kopal správným směrem.

Minecraft deník

Zapište si do deníku:

1. Jaký druh schodiště jste si vybrali? Rovný? Točitý? Diagonální? A Proč?
2. Na jaké problémy jste narazili? Jak jste je vyřešili?
3. Jak jste použili cykly při stavbě schodiště?
4. Popište případ, kde jste narazili na problém. Vysvětlete, jak jste dospěli k řešení.
5. Přidejte alespoň jeden snímek obrazovky vašeho schodiště.
6. Sdílejte váš projekt na webu a nezapomeňte vložit URL adresu.

Hodnocení

	1	2	3	4
Diář	V deníku chybí odpovědi na 4 nebo více otázek.	V deníku chybí 2 nebo 3 odpovědi.	V deníku chybí 1 odpověď.	Deník obsahuje všechny odpovědi.
Smyčky (Smyčky)	Nepoužívá vůbec smyčky nebo používá smyčky neužitečným způsobem.	Používá smyčky efektivně, ale povrchně.	Smyčky jsou nedílnou součástí programu, ale některé cykly nefungují správně.	Používá smyčky efektivním způsobem.
Projekt	Schodiště postrádá všechny požadované prvky.	Schodiště postrádá 2 požadované prvky.	Schodiště postrádá 1 požadovaný prvek.	Schodiště je: • Kompletní • Průchozí v obou směrech. • Končí v hloubce Y13

Podmínky

Počítačové programy jsou vytvořené z instrukcí, které říkají počítači, jak zpracovat vstup a jak doručit výstup. V Minecraftu (dále jen MC) jsou příkazy spuštěny buď pomocí příkazu, nebo tím, že nastane nějaká událost ve světě MC. Důležitou částí programování je přikazování počítači KDY má vykonat nějaký příkaz nebo úkon. K tomu používáme tzv. „podmínky“. Díky nim musí být splněna nějaká podmínka nebo provedena akce předtím, než se něco stane.

Žáci už jsou dávno seznámeni s konceptem podmínek z běžného života!

Určitě už slyšeli rodiče někdy říkat …

- „Pokud si uklidíš v pokoji, můžeš jít s kamarády ven.“
- „Pokud budeš mít hotový domácí úkol, můžeš hrát videohry.“
- „Pokud si budeš plnit své povinnosti, dostaneš povolení, jinak budeš mít „zaracha“.“

Tohle všechno jsou podmínky! Podmínky jsou ve formátu: POKUD něco, TAK/POTOM něco.
POKUD (podmínka je splněna), TAK/POTOM (stane se nějaká akce)

Nechte žáky, ať vymyslí pár dalších podmínek z jejich běžného života.

POZNÁMKA: Pro starší žáky můžete přidat do podmínky i část JINAK.

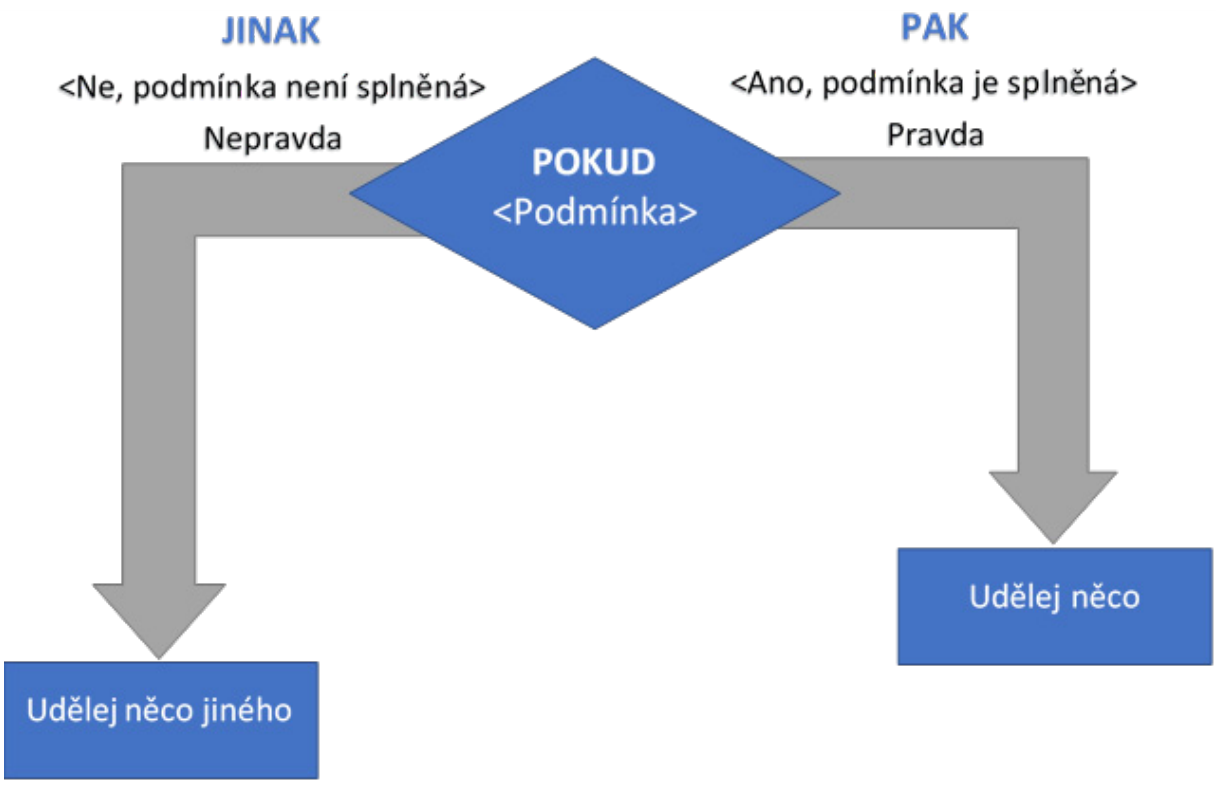
POKUD (podmínka je splněna), TAK/POTOM (stane se nějaká akce), JINAK (stane se něco jiného)

Příklady:

POKUD bude sněžit, TAK/POTOM si vezmi zimní boty, JINAK si obuj botasky.

Sekce JINAK zajistí, že se stane nějaká akce v každém případě.

Příklad: Pro větu: „POKUD bude sněžit, TAK/POTOM si vezmi zimní boty, JINAK si obuj botasky.“ Všimněte si části, JINAK si obuj botasky. Kdyby tam nebyla, mohlo by se stát, že žáci vyrazí na boso!



Offline: Simon říká, POKUD a až potom JINAK

Tahle aktivita je jen jednoduchou variací hry Simon říká na procvičení podmínek (Pokud, tak/potom, jinak)

Pravidla hry:

- Všichni si stoupnou.
- Učitel bude říkat sérii podmínkových vět (třeba: Pokud něco, tak něco. Některé mohou být: Pokud něco, tak něco, jinak něco.).
- Žáci se budou řídit pokyny.

Tady je jednoduchý příklad: Učitel řekne „Pokud máte hnědé vlasy, tak zvedněte svou pravou nohu, jinak si sedněte.“

Všichni s hnědými vlasy zvednou svou pravou nohu a zbytek si sedne.

Tipy pro učitele:

- Zadávejte jednoduché pokyny k procvičení s odlišnými pozicemi/úkony, aby byly výsledky dobře znatelné.
- Kontrolujte, zda žáci správně provedli pokyny dle zadání.
- Čtěte podmínkové věty pomalu a zřetelně.
- Buďte si jistí tím, že žáci pokynům rozumějí a reagují na všechny podmínkové věty.
- Mezi jednotlivými podmínkovými větami vracejte žáky do normálních pozic.
- Vyberte několik dobrovolníků, kteří vytvoří jejich vlastní podmínkové věty pro zbytek třídy.

Příklady některých podmínkových vět:

- Pokud vaše jméno končí na "J", dejte palce nahoru.
- Pokud měsíc vašeho narození začíná na "Č" nebo "S", zvedněte obě vaše ruce.
- Pokud hrajete fotbal, zvedněte jednu nohu.
- Pokud je vaše nejoblíbenější příchuť zmrzliny čokoláda, pak vyplázněte jazyk, jinak si udělejte králičí uši za hlavou.
- Pokud hrajete na nástroj, luskněte, jinak zapískejte.
- Pokud je váš nejoblíbenější mob v Minecraftu zombie, udělejte zvuky zombíka, jinak vyskočte a sedněte si.

Další příklady podmínek:

- Pokud jsou vaše vlasy hnědé/blond/černé
- Pokud máte černé/modré/červené oči
- Pokud máte na sobě svetr/bundu/tričko/sukni/sandály
- Pokud dnes prší/je slunečno/je oblačno
- Pokud jste za minulý měsíc hráli fotbal/volejbal/tenis
- Pokud jste jedli špagety/šunku/cereálie
- Pokud se vám dobře spalo/špatně se vám vstávalo
- Pokud je vaše oblíbená příchuť zmrzliny čokoládová/vanilková/karamelová
- Pokud máte pejska/kočku/andulku
- Pokud máte sestru/bratra/nemáte sourozence

Typy podmínek v MakeCode

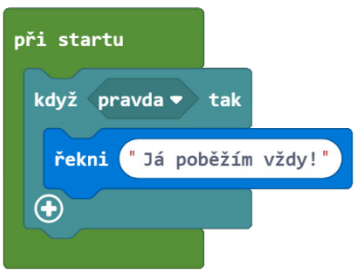
MakeCode pro MC obsahuje speciální blok nazvaný KDYŽ... TAK...

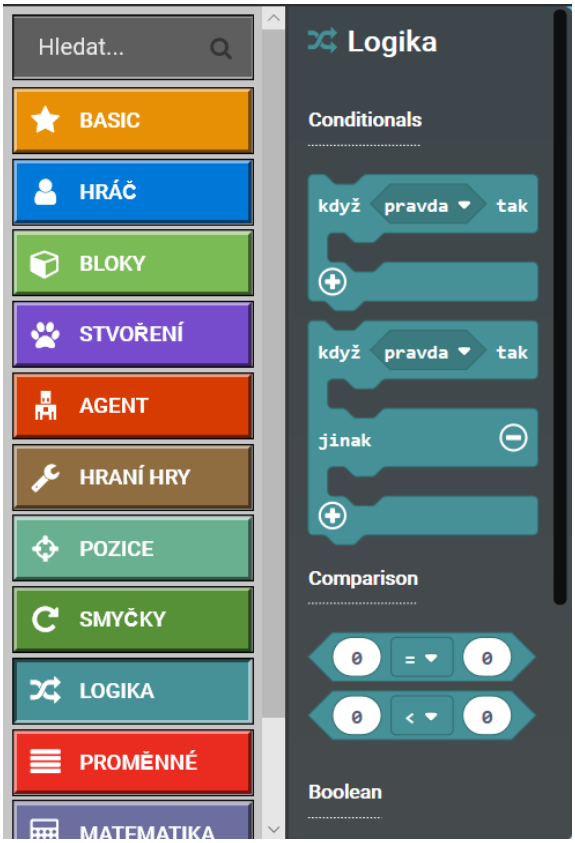
Dále obsahuje blok zvaný KDYŽ... TAK... JINAK...

Tyto bloky naleznete v nabídce Logika. Oba bloky kontrolují, zda je nějaká podmínka pravdivá a pokud ano, vykonají daný příkaz.

Pokud kliknete na malý symbol plusu v dolní části bloku, můžete přidat další podmínky.

V základu je vždy ve špičatém slotu blok „pravda“. Pokud ho nezměníte, vše, co se nachází v bloku Když...tak..., se vždy provede, protože podmínka „pravda“ je vždy pravdivá.





Místo špičatého bloku můžete vložit jiný blok nebo dokonce kombinaci bloků (na horním obrázku „pravda“), ale pouze v případě, že mají tyto bloky také špičaté resp. kulaté rohy. Všímejte si dobře tvarů bloků. Pomohou vám při sestavování vašeho kódu.

V nabídce Bloky můžete najít například tento špičatý blok:

Tento blok zjišťuje přítomnost určitého bloku na souřadnicích hráče. Výsledek tohoto testu bude vždy nepravda, jelikož na místě, kde stojíte, nemůže být žádný blok. Ale mohli byste zadat relativní souřadnice ~0 ~-1 ~0, aby se zjišťovala přítomnost bloku pod hráčem, nebo dokonce nějaký rozptyl souřadnic pro zjišťování přítomnosti bloku v dané oblasti.

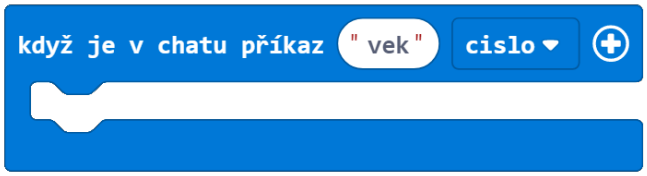
V této kapitole se budeme zabývat možnostmi, které různé podmínkové bloky poskytují. Například budeme automatizovat běžné činnosti jako třeba kácení stromů nebo těžbu surovin. Na závěr budete mít prostor vyzkoušet si použití podmínek ve vlastních projektech. Vzhůru na to!

Aktivita: Kolik ti je let?

V této programovací aktivitě si žáci procvičí práci s podmínkou Když Tak Jinak v MakeCode. Žáci si naprogramují program využívající podmínku k porovnání věků jich samých a jejich kamarádů. Výsledkem bude zpráva na obrazovce, zda jsou mladší, stejně staří nebo starší.

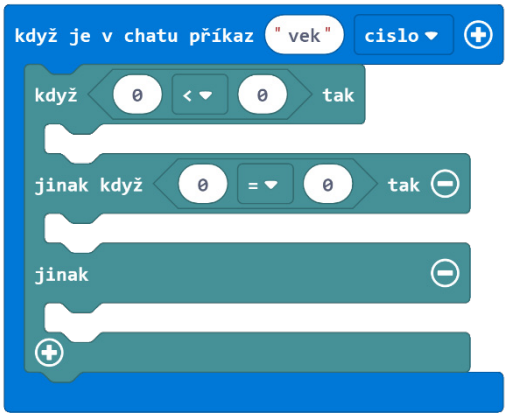
Jak na to:

1. Vytvořte nový projekt a nazvěte ho „Kolik ti je let“.
2. Přejmenujte blok Když je v chatu příkaz na "vek".
3. Klikněte na znaménko (+) a vytvořte nový proměnný parametr *num1* a přejmenujte ho na *cislo*.



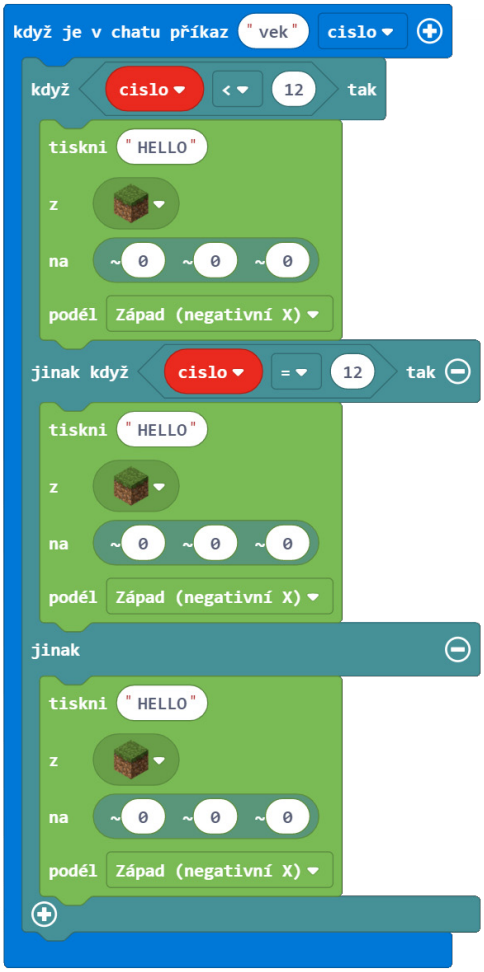
4. Z nabídky Logika přetáhněte Když Tak Jinak do bloku Když je v chatu příkaz.
5. Klikněte ne znaménko plus (+) u podmínkového bloku a přidejte Jinak Když.

6. Z nabídky Logika přetáhněte porovnávací blok Méně než a vložte ho do špičatého bloku „pravda“ za Když.
7. Znovu z nabídky Logika přetáhněte blok, ale tentokrát Rovná se (=) a vložte ho místo pravda za Jinak Když (viz obrázek)



8. Z nabídky Proměnné přetáhněte dvakrát blok cislo a vložte ho do prvního slotu do každého z porovnávacích bloků.
9. Do druhého slotu porovnávacích bloků zadejte váš věk (například 12).

10. Z nabídky Bloky přetáhněte tři bloky Tiskni. Můžete vzít pouze jeden a pravým tlačítkem myši ho kopírovat.
11. Dejte jeden do každého bloku Když...Jinak Když... Jinak...

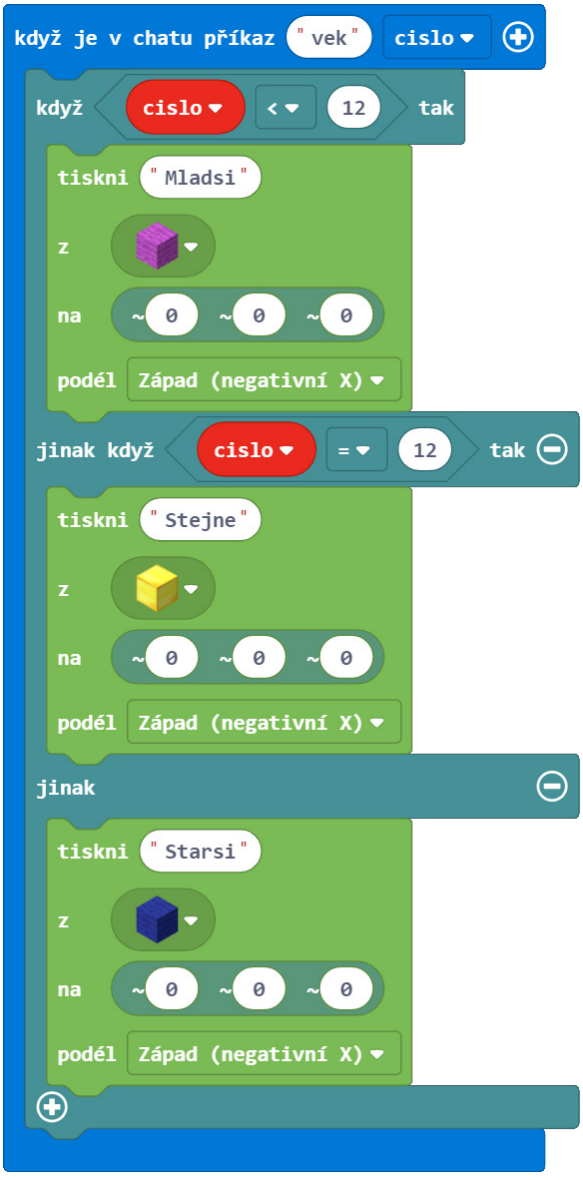


12. Do každého z bloků Tiskni napište různé zprávy pro žáky mladší, starší a stejně staré jako jste vy. Snažte se udržet zprávu krátkou – třeba jen 1 až 2 slova.
13. V rozbalovacím menu si vyberte různé bloky, kterými se bude nápis tisknout.

Nechte žáky, ať vytvoří dvojice a vyzkouší si navzájem své programy:

- Stiskněte klávesu „T“, otevře se chatovací okno.
- Druhý z dvojice napíše příkaz *vek x*, kde x je věk hráče.
- Hráč musí zůstat nehybně na své pozici, dokud se nenapíše celá zpráva.
- Na závěr poodstupte a můžete si prohlédnout výsledek.

Kompletní program:



JavaScript:

```
player.onChat("vek", function (cislo) {
    let cislo = 0
    if (cislo < 12) {
        blocks.print(
        "Mladsi",
        MAGENTA_WOOL,
        pos(0, 0, 0),
        WEST
        )
    } else if (cislo == 12) {
        blocks.print(
        "Stejne",
        GOLD_BLOCK,
        pos(0, 0, 0),
        WEST
        )
    } else {
        blocks.print(
        "Starsi",
        BLUE_WOOL,
        pos(0, 0, 0),
        WEST
        )
    }
})
```

Sdílený program: https://makecode.com/_4R5VTf0bD4AM

Aktivita: Dřevorubec

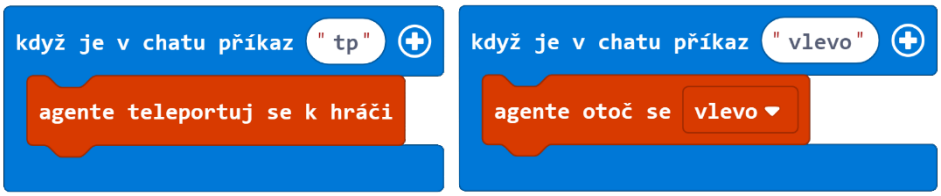
Sekání stromů kvůli dřevu je těžké, ale nutné, pokud chcete v režimu Přežití něco vyrobit. S pomocí Agenta můžeme tuto fušku zautomatizovat! Pojďme Agenta naučit, jak pokácet strom jakékoli výšky.

Když začnete nový projekt v aplikaci MakeCode, je často dobré naplánovat, jaké kroky chcete udělat ještě před samotným začátkem kódování. Pojďme si promyslet, co chceme udělat:

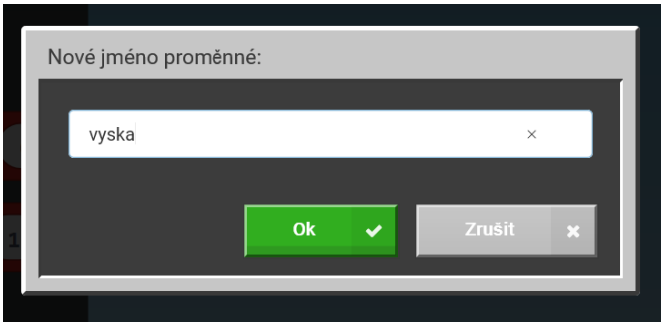
1. Zjistit velikost stromu:
   - Dokud je před Agentem blok, pohybuj se nahoru
   - Zaznamenávej výšku stromu
2. Opakovat tolikrát, jako je výška stromu:
   - Znič blok před sebou
   - Posuň se dolů
3. Vše posbírat

Kroky:

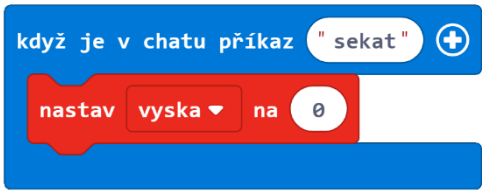
1. Vytvořte nový projekt „Dřevorubec“.
2. Přejmenujte existující Když je v chatu příkaz "tp".
3. Z nabídky Agent v panelu nástrojů přetáhněte blok Agente teleportuj se k hráči a umístěte ho do Když je v chatu příkaz "tp".
   Teď pojďme vytvořit směrový příkaz, abychom mohli Agentem otáčet.
4. Z nabídky Hráč přetáhněte blok Když je v chatu příkaz do pracovního prostoru.
5. Přejmenujte tento Když je v chatu příkaz na "vlevo".
6. Z nabídky Agent přetáhněte blok Agente otoč se a vložte jej do Když je v chatu příkaz "vlevo".



7. Z nabídky Hráč přetáhněte blok Když je v chatu příkaz a přejmenujte jej na "sekat".
   Pro uložení výšky stromu využijeme proměnnou.
8. V nabídce Proměnné klikněte na tlačítko „Vytvořit proměnnou“.
9. Přejmenujte proměnnou na „vyska“.



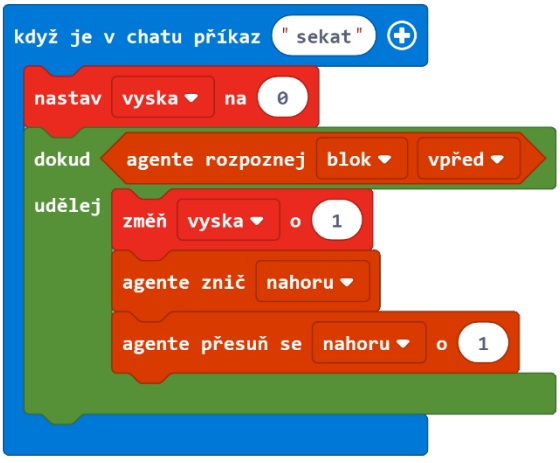
10. Z nabídky Proměnné přetáhněte blok Nastav a umístěte ho pod blok Když je v chatu příkaz "sekat".
11. V bloku Nastav vyberte z rozbalovací nabídky proměnnou „vyska“.



Budeme předpokládat, že při zadání rozkazu stojí Agent na zemi čelem ke kmeni stromu. Použijme cyklus Dokud, který funguje jako opakovací cyklus kombinovaný s podmínkou, abychom zjistili, jestli je před Agentem blok, a tudíž mohli šplhat nahoru po stromě.

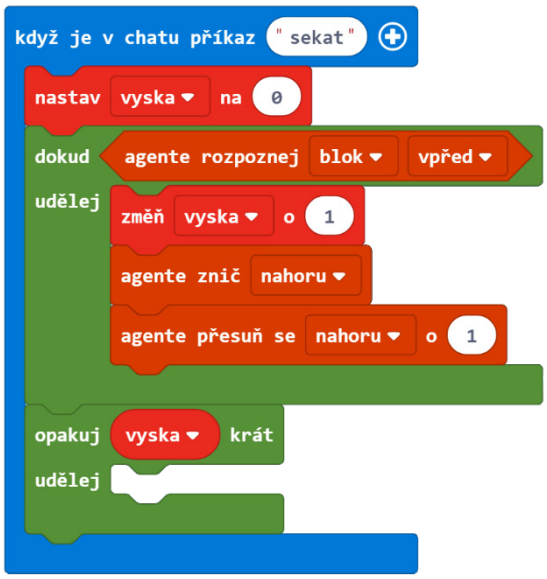
12. Z nabídky Smyčky přetáhněte cyklus Dokud a vložte ho pod blok Nastav v Když je v chatu příkaz "sekat".
13. Z nabídky Agent přetáhněte blok Agente rozpoznej a vložte ho dovnitř cyklu Dokud místo bloku Pravda.

14. Z nabídky Proměnné přetáhněte blok Změň a vložte ho dovnitř cyklu Dokud.
15. V bloku Změň vyberte z rozbalovací nabídky proměnnou „vyska“.
16. Tento blok bude zvyšovat proměnnou vyska pokaždé, kdy Agent postoupí vzhůru, takže můžeme zaznamenat výšku stromu.
17. Z nabídky Agent přetáhněte bloky Agente znič a Agente přesuň se a oba umístěte pod blok Změň v našem cyklu Dokud.
18. V bloku Agente znič vyberte z rozbalovací nabídky směr „nahoru“.
19. V bloku Agente přesuň se vyberte z rozbalovací nabídky směr „nahoru“. V případě, že jsou nad Agentovo hlavou listy nebo větve, budeme se snažit je zničit, abychom dostali Agenta k vrcholu kmene.



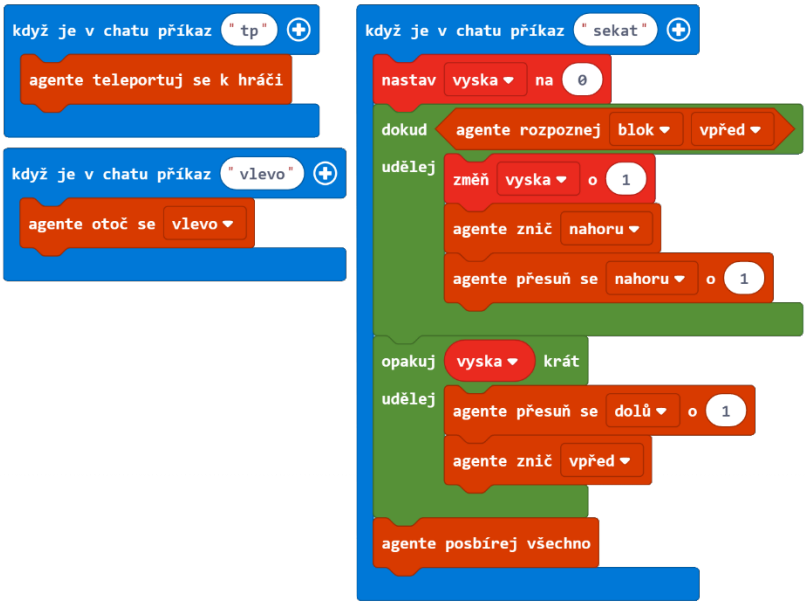
Jakmile Agent dosáhne vrcholku stromu (nad ním už není blok), je výška stromu uložena v proměnné vyska. Budeme potřebovat ještě jeden cyklus, který zničí blok před Agentem (kmen stromu) a posune Agenta dolů. Tento cyklus by se měl spustit tolikrát, jako je uložená výška stromu v proměnné vyska.

20. Z nabídky Smyčky vyberte cyklus Opakuj a umístěte jej pod cyklus Dokud v našem Když je v chatu příkaz "sekat".
21. Z nabídky Proměnné přetáhněte blok s proměnnou vyska a vložte jej dovnitř cyklu Opakuj místo čísla 4.



22. Z nabídky Agent přetáhněte bloky Agente přesuň se a Agente znič a v tomto pořadí je umístěte pod cyklus Opakuj.
23. V bloku Agente přesuň se vyberte z rozbalovací nabídky směr „dolů“.
    Pamatujte, že jsme se zastavili, když před námi už nebyl blok, takže v tomto cyklu se nejdřív potřebujeme posunout dolů a až pak ničit dopředu. Konečně, když se Agent dostane na zem, chceme se ujistit, že posbírá všechno dřevo, které vytěžil.
24. Z nabídky Agent přetáhněte blok Agente posbírej všechno a vložte ho za cyklus Opakuj jako poslední část našeho příkazu.

Kompletní program:



JavaScript:

```
let vyska = 0
player.onChat("vlevo", function () {
    agent.turn(LEFT_TURN)
})
player.onChat("sekat", function () {
    vyska = 0
    while (agent.detect(AgentDetection.Block, FORWARD))
{
        vyska += 1
        agent.destroy(UP)
        agent.move(UP, 1)
    }
    for (let index = 0; index < vyska; index++) {
        agent.move(DOWN, 1)
        agent.destroy(FORWARD)
    }
    agent.collectAll()
})
player.onChat("tp", function () {
    agent.teleportToPlayer()
})
```

Sdílený program: https://makecode.com/_1hFLi6X1qPcJ

Rozšíření:

Je těžké předpokládat, kterým směrem se bude Agent dívat, když ho přivoláte. Proto se hodí směrové příkazy. Ale zvládnete to bez nich?

Můžete vynechat příkazy "tp" a "vlevo" a vše zahrnout v programu "sekat" následovně:

- Zařiďte, ať je Agente teleportuj se k hráči první příkaz, poté co napíšete do chatu „sekat“ a stojíte u základny stromu.
- Dokud se Agent NEDÍVÁ na blok, ať se stále otáčí vlevo. Zkuste zařídit, aby se Agent zastavil, jakmile se bude dívat na kmen stromu, pak pokračujte v programu podle postupu uvedeného na začátku.

Jediná možná nevýhoda tohoto plánu je, že pokud nemáte příkaz „vpred“, musíte při spuštění programu stát přímo u stromu. To znamená, že budete občas muset odstranit nízko visící listy. Ale možná přijdete na to, jak se tomu vyhnout!

Aktivita: Všechno moje!

Kromě kácení stromů pro dřevo budete pravděpodobně chtít těžit v podzemí vzácné minerály. Dobrá strategie pro hledání toho nejvzácnějšího (diamantu) je kopat tunel hluboko pod zemí na úrovni Y12 nebo Y13 a pak si vytvořit sérii větvících se tunelů z hlavní chodby. Pokud jste dokončili samostatný projekt: „Schodiště“ v předchozí lekci, máte pravděpodobně kód, díky kterému bude Agent těžit za vás. Pokud ne, dostaňte se dolů na úroveň Y12 nebo Y13 a uvidíme se tam! Nápověda – zjistěte svou pozici ve hře (v nastavení si zapněte zobrazení souřadnic nebo napište do chatu příkaz /gamerule showcoordinates true), zadejte příkaz */tp X 12 Z*, přičemž X a Z jsou vaše aktuální světové souřadnice v Minecraftu. Toto vás teleportuje rovnou dolů na úroveň 12. Ujistěte se, že s sebou máte krumpáč a louče, abyste si dole mohli vykopat prostor a rozsvítit!

V této aktivitě vytvoříme základního těžícího Agenta, který automaticky ničí bloky před sebou a sbírá vše okolo sebe. Využijeme podmíněné příkazy, abychom našli vzácné minerály. Agent je vysoký jeden blok, takže ho můžeme poslat kupředu 64 bloků, pak mu nařídit, ať se otočí, posune nahoru o blok a vrátí se zpět. Pokud Agent najde cokoli cenného, dá nám vědět a sebere danou rudu.

Kroky:

1. Vytvořte nový projekt v MakeCode a pojmenujte ho „Vytěž všechno“.
2. Přejmenujte existující Když je v chatu příkaz "tp".
3. Z nabídky Agent na panelu nástrojů přetáhněte blok Agente teleportuj se k hráči a umístěte ho pod Když je v chatu příkaz "tp".
4. Opět vytvoříme směrové příkazy, abychom mohli Agentem otáčet.
5. Z nabídky Hráč přetáhněte na pracovní plochu Když je v chatu příkaz.
6. Přejmenujte tento blok na Když je v chatu příkaz "vlevo".
7. Z nabídky Agent přetáhněte blok Agente otoč se a vložte jej do Když je v chatu příkaz "vlevo".

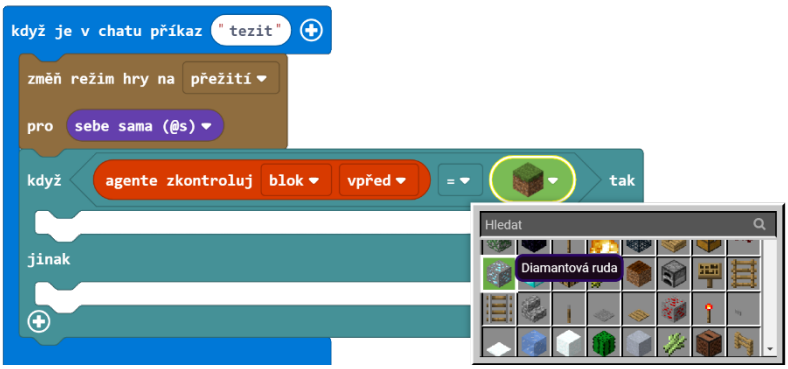
8. Z nabídky Hráč přetáhněte blok Když je v chatu příkaz do pracovního prostoru a přejmenujte ho na "tezit".
   Těžit diamanty má smysl pouze v režimu Přežití, proto si herní mód přepneme na Přežití.
9. Ze záložky Hraní hry přetáhněte blok Změň režim hry na a vložte ho do bloku Když je v chatu příkaz "tezit".
10. V bloku Změň režim hry na vyberte z rozbalovací nabídky "sebe sama @s".

11. Z nabídky Logika vyberte blok Když Tak Jinak a vložte jej pod blok Změň režim hry na.

Všimněte si, že podmínka pravda je blok se špičatými konci. Jak už jsme zmínili dříve, můžete vytvořit své vlastní podmínky, které musí být splněny, aby se bloky uvnitř podmínky spustily. V našem případě chceme podmínku: Když Agent najde hodnotné bloky jako zlatou nebo diamantovou rudu.

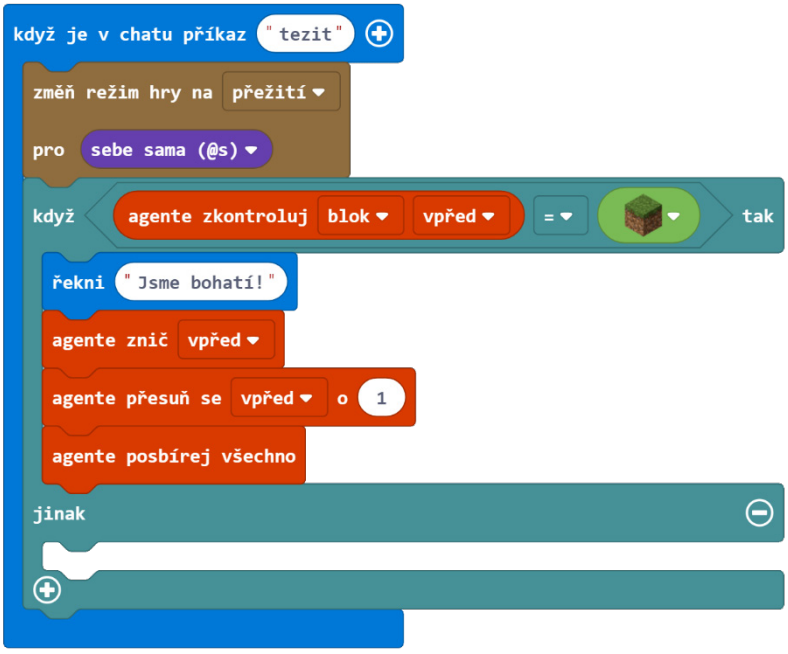
12. Z nabídky Logika přetáhněte blok Rovná se (=) a vložte ho dovnitř bloku Když Tak Jinak namísto "pravda".
13. Z nabídky Agent přetáhněte blok Agente zkontroluj do prvního slotu v bloku Rovná se (=) místo čísla 0. Blok Agente zkontroluj zkontroluje blok přesně před Agentem.

14. Z nabídky Bloky přetáhněte blok Blok a vložte ho do druhého slotu bloku Rovná se (=) místo čísla 0.
15. V bloku Blok vyberte z rozbalovací nabídky kostku „Diamantová ruda“, kterou hledáme (k rychlejšímu nalezení konkrétního bloku v inventáři můžete využít vyhledávací řádek).



Když najdeme diamantovou rudu, chceme na obrazovce vypsat zprávu a daný blok vytěžit.

16. Z nabídky Hráč v panelu nástrojů přetáhněte blok Řekni do bloku Když Tak.
17. V bloku Řekni napište zprávu ve smyslu „Jsme bohatí!“.
18. Z nabídky Agent přetáhněte následující 3 bloky a vložte je pod blok Řekni: Agente znič, Agente přesuň se a Agente posbírej všechno.



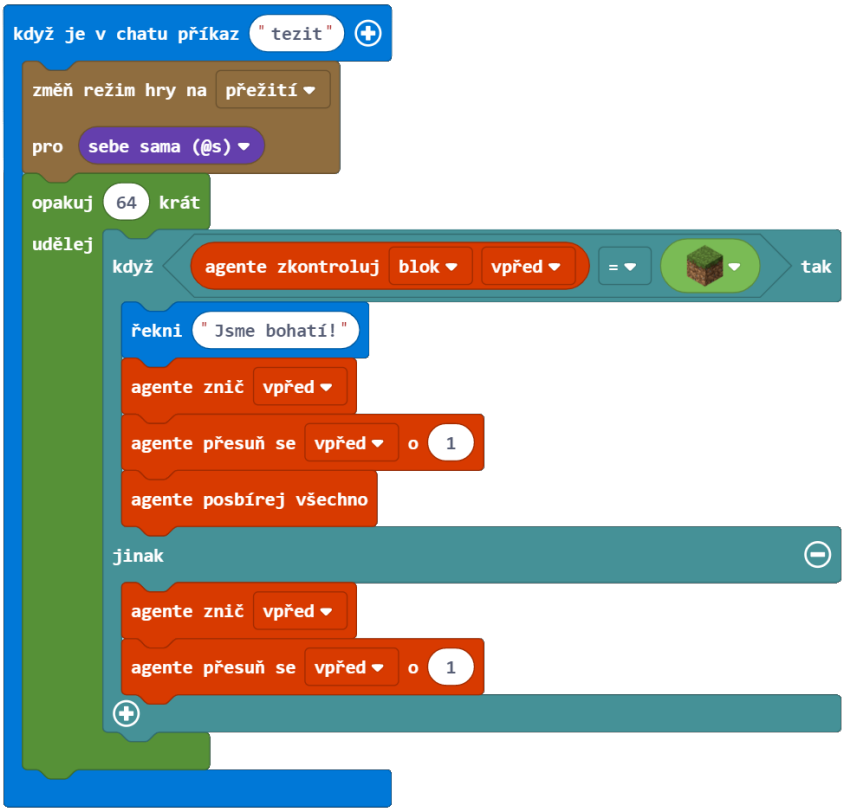
V případě, že Agent nemá před sebou diamanty, jednoduše by měl zničit bezcenný blok před sebou a posunout se kupředu, aniž by něco sbíral.

19. Z karty Agent přetáhněte bloky Agente znič a Agente přesuň se a vložte je do možnosti Jinak v bloku Když Tak Jinak.

Nyní zopakujeme celý proces 64krát. K tomu využijeme cyklus.

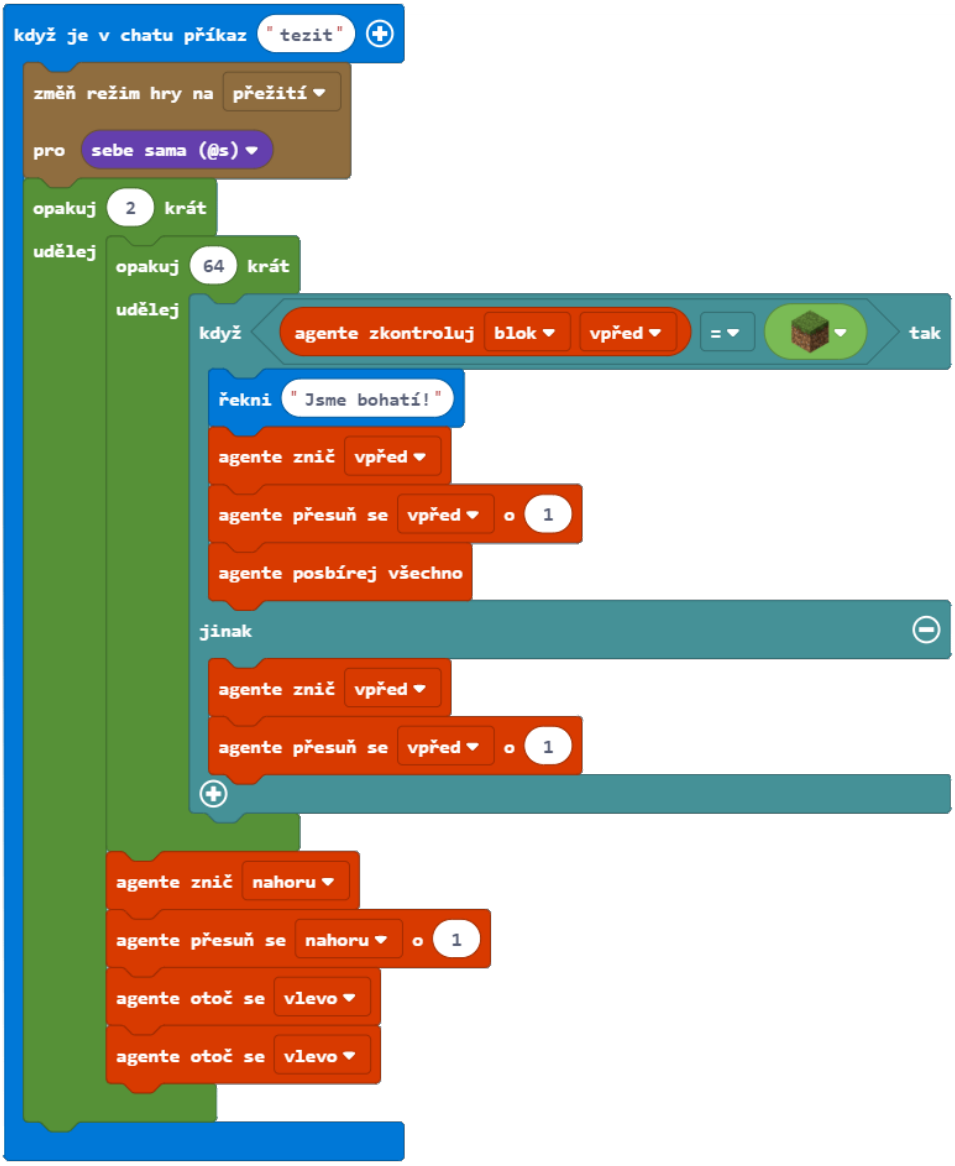
20. Z nabídky Smyčky přetáhněte cyklus Opakuj, umístěte ho za blok Změň režim hry na a blok Když Tak Jinak přesuňte dovnitř tohoto cyklu.

21. V cyklu Opakuj zadejte počet opakování 64.



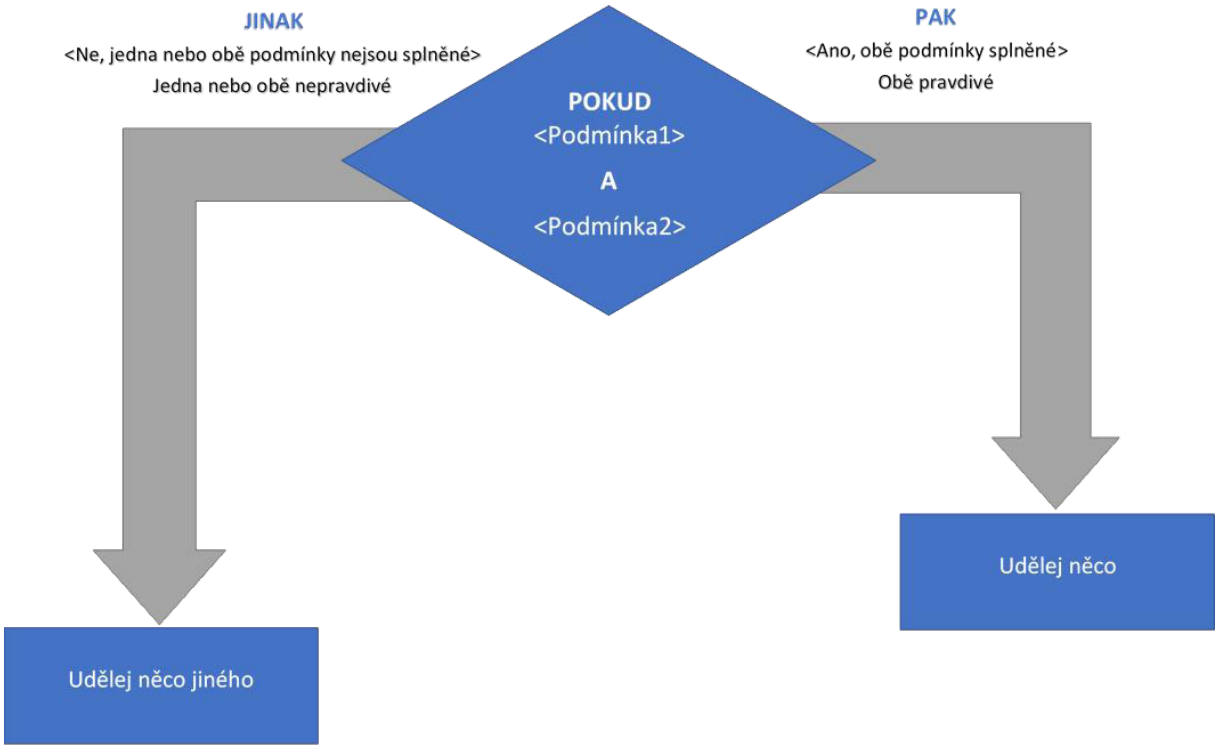
Tímto pošleme našeho Agenta dopředu, aby vytěžil 64 bloků. Pro prohledání větší oblasti potřebujeme, aby se Agent otočil a vrátil. Proto využijeme další cyklus okolo našeho cyklu Opakuj – tomu se říká vložený cyklus.

22. Z nabídky Smyčky přetáhněte další cyklus Opakuj a vložte ho pod blok Změň režim hry na a obklopte stávající cyklus Opakuj 64 tímto cyklem Opakuj. Zadejte počet opakování 2, protože chceme, aby náš Agent vytěžil jen dvě řady bloků. (tato čísla můžete libovolně upravovat a měnit tak velikost prohledávané oblasti).
    Na konci první řady chceme, aby se náš Agent posunul nahoru a otočil se.
23. Z nabídky Agent přetáhněte následující 4 bloky a v tomto pořadí je vložte do vnitřního Opakuj cyklu: Agente znič, Agente přesuň se, a 2 bloky Agente otoč se.
24. V bloku Agente znič vyberte z rozbalovací nabídky směr „nahoru“.
25. V bloku Agente přesuň se vyberte z rozbalovací nabídky směr „nahoru“.

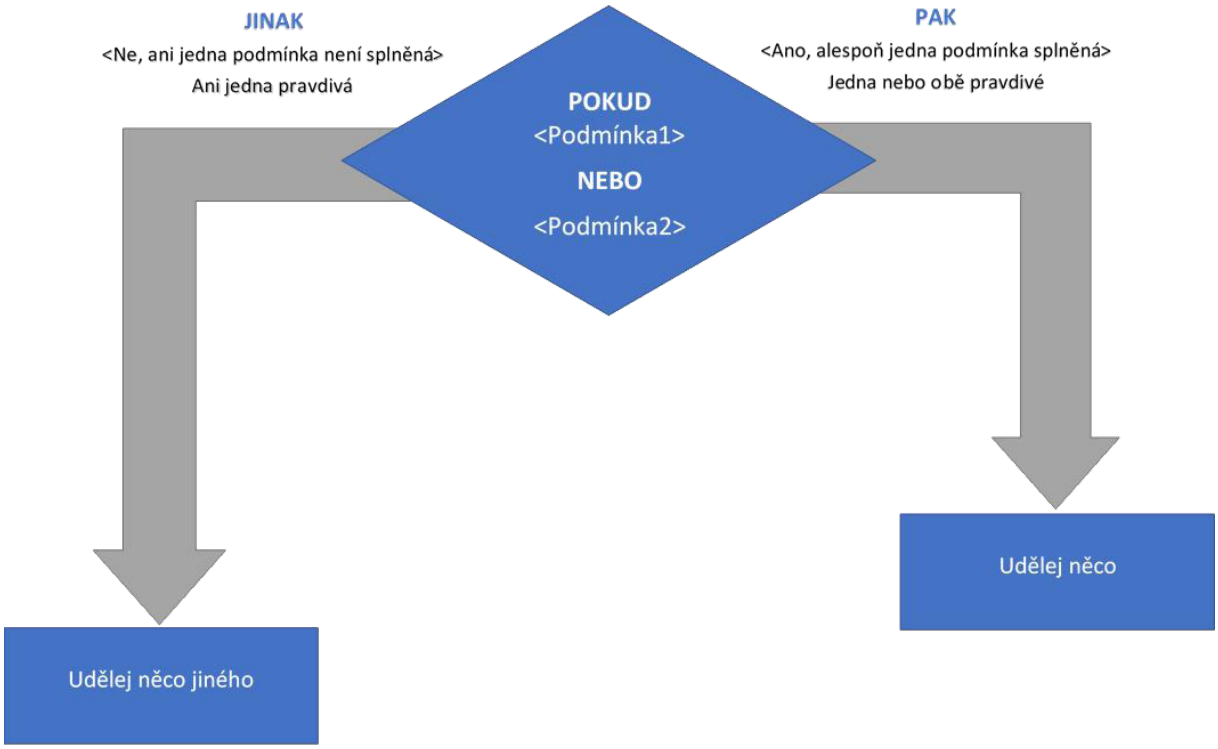


Dobrovolné rozšíření:

Další hodnotný minerál, který najdete na této úrovni, je zlato. Do našeho programu můžeme kromě kontroly diamantů přidat i zlato. K vytvoření kódu musíme využít logický operátor NEBO. MakeCode pro Minecraft podporuje logické (booleovské) operace. Dík nim můžeme vytvářet složitější podmínky s využitím operátorů NEBO (pravdivý v případě, když je alespoň jedna podmínka pravdivá) nebo A (pravdivý v případě, kdy jsou obě podmínky zároveň pravdivé).



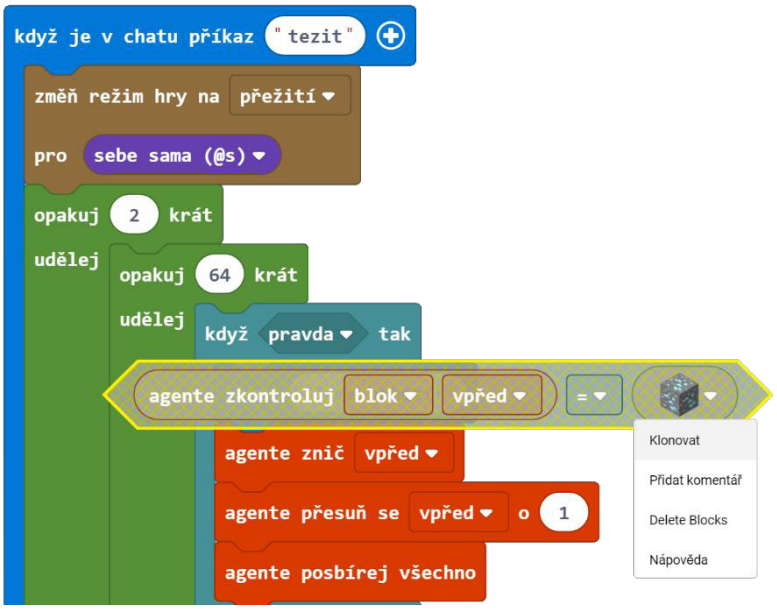
AND operátor: obě podmínky musejí být splněné



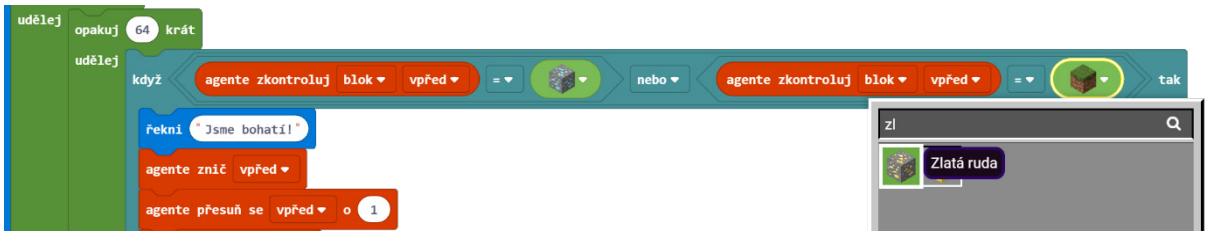
OR (NEBO) operátor: alespoň jedna podmínka musí být splněna

26. Z nabídky Logika přetáhněte blok Nebo do bloku Když Tak Jinak místo porovnávacího bloku Rovná se.

27. Porovnávací blok Rovná se se bude vznášet odpojený a zašedlý ve vašem pracovním prostoru. Klikněte na něj pravým tlačítkem myši a vyberte Klonovat pro vytvoření další kopie.

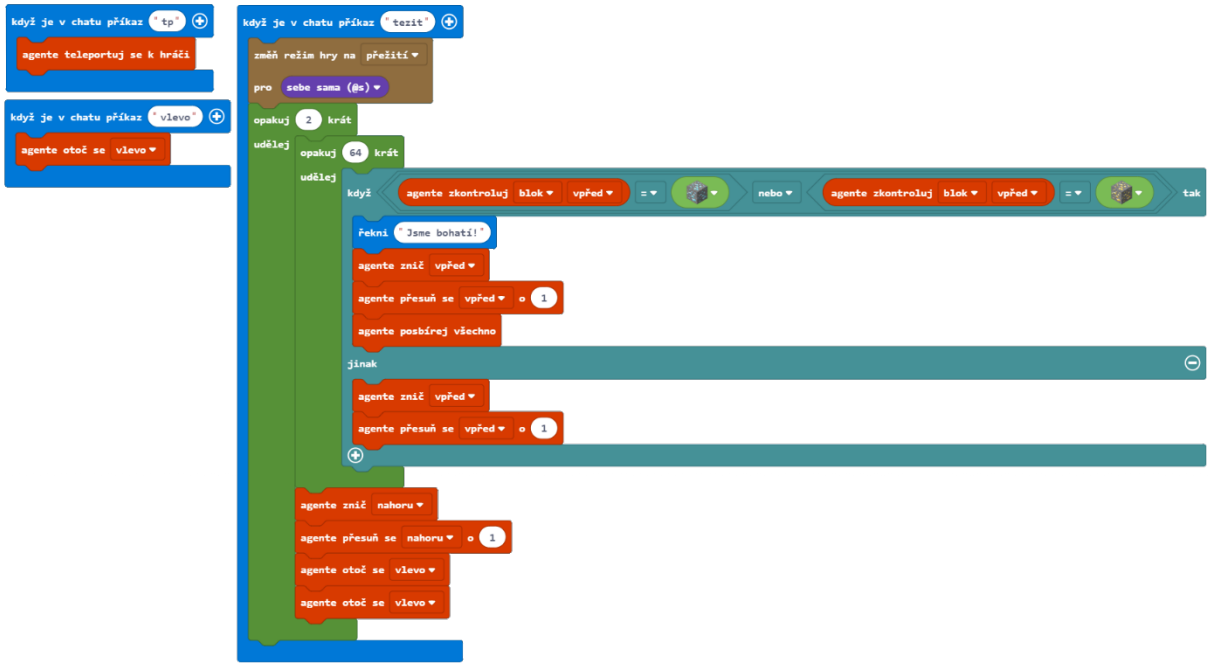


28. Přetáhněte oba porovnávací bloky Rovná se do prvního a druhého políčka bloku NEBO.
29. V druhém porovnávacím bloku Rovná se vyberte z rozbalovací nabídky Blok a jako hledaný minerál zvolte „Zlatá ruda“.



V tomto případě se pokyn Když spustí, pouze pokud bude před Agentem diamantová nebo zlatá ruda. Pokud je *jedna* z těchto podmínek pravda, vypíšeme na obrazovku „Jsme bohatí!“, zničíme blok před námi (z něj vypadnou diamanty nebo zlatá ruda), posuneme se dopředu a sebereme drahokamy. Jinak Agent jednoduše zničí bezcenný blok před sebou a posune se dopředu, aniž by cokoli sbíral.

Kompletní kód:



JavaScript:

```
player.onChat("tezit", function () {
    gameplay.setGameMode(
    SURVIVAL,
    mobs.target(LOCAL_PLAYER)
    )
    for (let index = 0; index < 2; index++) {
        for (let index = 0; index < 64; index++) {
            if (agent.inspect(AgentInspection.Block, FOR
WARD) == DIAMOND_ORE || agent.inspect(AgentInspection.Bl
ock, FORWARD) == GOLD_ORE) {
                player.say("Jsme bohatí!")
                agent.destroy(FORWARD)
                agent.move(FORWARD, 1)
                agent.collectAll()
            } else {
                agent.destroy(FORWARD)
                agent.move(FORWARD, 1)
            }
        }
        agent.destroy(UP)
```

```
        agent.move(UP, 1)
        agent.turn(LEFT_TURN)
        agent.turn(LEFT_TURN)
    }
})
player.onChat("vlevo", function () {
    agent.turn(LEFT_TURN)
})
player.onChat("tp", function () {
    agent.teleportToPlayer()
})
```

Sdílený program: https://makecode.com/_8Aif41bHEdFo

Skupinový projekt: Podmínky

V této kapitole jsme se podívali na to, jak fungují různé podmínkové bloky. Naučili jsme, že cyklus Dokud je v podstatě cyklus Opakuj v kombinaci s podmíněným příkazem (opakuj, dokud je splněna podmínka) a také jsme se naučili, jak můžeme využít sérii podmínek Když oddělených částmi Jinak, abychom vytvořili bloky kódu provádějící všechny možné operace. A nakonec jsme se naučili pracovat s operátory se dvěma možnostmi (booleovské operátory), které popisují vztah mezi podmínkami, např.: obě podmínky musí být splněné nebo alespoň jedna musí být splněna, aby byl celý výraz platný.

V tomto samostatném projektu pracujte se spolužákem, abyste spolu vytvořili projekt, který využívá jeden nebo více bloků ze záložky Logika. Přemýšlejte o problémech, se kterými se setkáte ve světě Minecraftu a pokuste se pro ně vymyslet řešení.

Techniky práce ve dvojicích

Spolupráce na návrhu

V jakémkoli projektu je důležité začít pochopením problému. Tuto aktivitu můžete začít rozhovory s lidmi okolo vás, kteří se setkali s problémem, který se snažíte vyřešit. Například pokud navrhujete nástroj pro přežití, co dělá většina hráčů Minecraftu, aby přežila první noc? Zeptejte se svých spolužáků, jak shánějí jídlo, staví přístřeší atd. Jaké zdroje hledají obvykle první? Mají systém? Co by bylo účinnější? Co by si přáli být schopní udělat?

Pokud navrhujete něco jiného, přemýšlejte, jak zjistit více informací o svém problému prostřednictvím rozhovorů nebo pozorování lidí, jak hrají Minecraft. Soustřeďte se přímo na jejich chování ve hře. Dobrý způsob pro přesné pozorování je předem si sestavit seznam klíčových otázek:

Co dělají noví hráči jako první?

Jak rychle se rozhodnou jít pod zem?

Jak dlouho tam zůstanou?

Mají ve zvyku zůstat na jednom místě, nebo se stěhovat?

Jaké vlastnosti má mít dobré místo pro stavbu základny?

Pak si se svým spolužákem vyzkoušejte brainstorming. Mluvte o široké škále různých nápadů. Pamatujte, že nevadí, když nápady působí nelogicky nebo neprakticky. Některé z nejlepších programů pochází ze zdánlivě šílených nápadů, které se dají nakonec přepracovat v užitečný program.

Programování ve dvojicích

Koncept programování ve dvojici je cenný způsob, jakým programátoři spolupracují při společném tvoření programů. Dva programátoři sdílejí jeden počítač, přičemž je jedna osoba u klávesnice jako řidič a druhá osoba udává směr jako navigátor. Pokud pracujete na projektu s někým jiným, zkuste používat jeden počítač a střídat se při tvoření kódu a pak se podívat, jak se chová ve světě Minecraftu. Hlavním pravidlem je, že pokud jste u klávesnice, děláte to, co vám řekne druhý člověk. On je navigátor! Pamatujte na procvičování dobré vzájemné komunikace po celý proces programování.

Debugging (ladění programu)

Určitě zkuste všechny možné nápady. Jaký je nejjednodušší způsob, jak si udržovat přehled o datech? Můžete zkusit, jestli jsou jisté podmínky detekovány přidáním bloku Řekni dovnitř bloku Když Tak. Pokud ve světě uděláte něco, co má spustit podmínku, ale nevidíte na obrazovce zprávu, víte, že je s vaší podmínkou něco špatně.

Podobně, pokud aktualizujete své proměnné v závislosti na jistých podmínkách, můžete použít blok Řekni pro vypsání hodnoty proměnné, abyste se ujistili, že se správně mění. Pokud využíváte číselné proměnné, nezapomeňte použít Spoj blok, abyste je připojili ke slovu (nebo prázdným uvozovkám) a tím si převedli číselnou hodnotu na text.

Zde jsou příklady výzev pro váš samostatný projekt:

Hledač železa

Jeden z nejběžnějších úkolů na začátku hry je najít dostatek železa na výrobu železných nástrojů a možná i plné sady železného brnění na ochranu. Železo se nachází v žílách na kamenných svazích a v jeskyních, ale často není vidět na první pohled. Vytvořte projekt v MakeCode, který zjistí přítomnost železa v okolí a informuje vás, když se k železné rudě přiblížíte. Jakmile najdete jeden blok železné rudy, pravděpodobně jich bude v okolí víc, takže vám bohatě stačí rozbít jeden blok a máte železnou žílu!

Vylepšený horník

Dokážete vylepšit již vytvořený program? Dokážete přimět Agenta, aby kontroloval více úrovní zároveň? Dokážete hledat další vzácné minerály jako třeba smaragd?

Minecraft deník

Zapište si do deníku:

- Jaký problém v Minecraftu jste se rozhodli vyřešit? Co dělá váš program?
- Jak jste ve vašem projektu využili podmíněné příkazy?
- Diskutujte o jednom (nebo více) směrech, jak byla práce se spolužákem jiná než řešení na vlastní pěst.
- Popište jeden okamžik, kde jste se „zasekli“. Pak popište, jak jste ho vyřešili.
- Zahrňte alespoň jeden snímek s vaším Agentem v akci.
- Sdílejte svůj projekt na webu a přidejte sem URL odkaz.

POZNÁMKA: Pokud jste se rozhodli zlepšit jednu z aktivit z této lekce, prosím, diskutujte o novém kódu, který jste napsali jako vylepšení k již poskytnutému základu.

Hodnocení

	1	2	3	4
Deník	V deníku chybí odpovědi na 4 nebo více otázek.	V deníku chybí odpovědi na 2 nebo 3 otázky.	V deníku chybí odpovědi na 1 otázku.	Deník obsahuje všechny odpovědi.
Logika	Vůbec nevyužívá podmíněné příkazy.	Používá podmíněné příkazy efektivně ale povrchně.	Podmíněné příkazy jsou nedílnou součástí programu, ale má problémy s jejich vykonáváním.	Používá podmíněné příkazy efektivně.
Projekt	Projektu schází všechny požadované prvky.	Projektu schází 2 z požadovaných prvků.	Projektu chybí 1 z žádaných prvků.	Projekt řeší specifický problém účinně.

Funkce

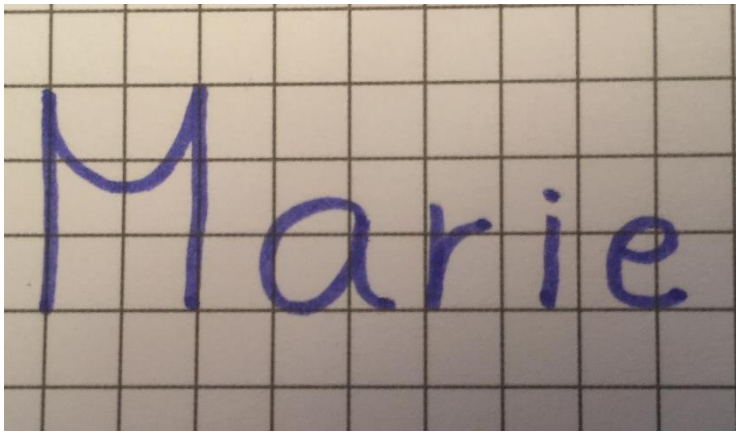
V programování jsou často úkoly či postupy, které se v rámci stejného programu opakují. Místo přepisování toho samého kódu, který provádí daný úkol pokaždé, když ho potřebujete, můžete shromáždit skupinu těchto instrukcí dohromady v jednu funkci. Shromažďování často užívaných instrukcí dělá váš kód efektivnější. Skupinu instrukcí můžete zapsat jako funkci a tuto funkci poté použít (zavolat) kdykoliv, kdy budete potřebovat. Funkce se většinou pojmenovává podle úkolu, který má vykonat. Tím dělá kód čitelnější.

Vlastní jméno

Představte si tento příklad: Jako žák či učitel pravděpodobně musíte napsat své jméno několikrát denně na papír, formulář či přihlašovací list. Napsání vašeho jména je funkce, kterou děláte bez složitého přemýšlení. Přesto je tato funkce složena z několika malých kroků. Podívejme se, jak by tyto kroky vypadaly pro osobu jménem 'Marie'. Tato funkce by se mohla jmenovat 'NapisMeJmeno'.

Seznam kroků či řádků kódu uložené ve funkci 'NapisMeJmeno' pro jméno Marie by vypadal takto:

- Napiš velké 'M'
- Hned napravo vedle předchozího písmena napiš malé 'a'
- Hned napravo vedle předchozího písmena napiš malé 'r'
- Hned napravo vedle předchozího písmena napiš malé 'i'
- Hned napravo vedle předchozího písmena napiš malé 'e'



Pokaždé, když bude Marie potřebovat napsat své jméno, zadá kód 'NapisMeJmeno' a provede se tato skupina instrukcí.

Pamatujte, že řádky pseudokódu této funkce mají mezi sebou ještě menší funkce.

Když jste se učili, jak napsat své jméno, potřebovali jste provést sadu instrukcí, abyste vůbec zvládli napsat každé písmeno.

NapisVelkeM, NapisMaleA, NapisMaleR, NapisMaleI a NapisMaleE jsou také funkce, které musely být provedeny, abyste mohli napsat požadovaný text.

Které skupiny instrukcí vytvoří každou z těchto menších funkcí?

Offline: Toast

Pojďme si ukázat další příklad, jak mohou být funkce použity k řešení úkolů.

Představte si, že vám rodič dělá každý den svačinu – nebylo by hezké, kdyby za něj mohl tento úkol dělat počítač?

Nechte žáky, aby si představili robota, který jim bude každý den dělat toast s máslem a džemem.

- Jaké kroky jsou potřeba k vytvoření takového toastu?
- Dokážete vytvořit funkci zahrnující všechny potřebné kroky k vytvoření chutného toastu?

Tato aktivita je k práci ve třídě jako stvořená. Ukažte ji žákům, ale neprozraďte jim, jak budete postupovat.

Materiály:

- Máslo nebo tavený sýr (uzavřené)
- Sklenice džemu (uzavřená)
- Krájený toastový chléb (zabalený)
- Talíř
- Nůž
- Papírové ubrousky

Instrukce:

Nechte žáky ve dvojicích napsat pseudokód pro vytvoření toastu s máslem a džemem.

Poté postupujte následovně:

- Vy, jako učitel, budete předstírat, že jste robot.
- Vyberte všechny pseudokódy od žáků a jeden z nich si vyberte k předvedení. Ten s nejméně kroky je k předvedení na začátek většinou nejlepší, jelikož žáci, kteří ho vytvořili, předpokládali, že jako robot již budete něco umět.
- Postupujte přesně podle instrukcí. Tohle je šance se trochu pobavit. Tím, že budete dělat přesně to, co je v instrukcích a ne to, co oni předpokládali, že uděláte.
    - Prvním krokem je často „Otevři pytlík s chlebem“. Klidně prostě pytlík s chlebem roztrhněte, neboť jejich instrukce neříkají nic o rozvázání nebo rozepnutí pytlíku s chlebem.
    - Pokud je instrukce „Polož na talíř dva krajíce chleba“, klidně tyto dva krajíce položte na sebe, neboť jejich instrukce se nezmiňují o tom, jak by měly být na talíř položeny.
    - Pokud je jejich další instrukce „Namaž jeden krajíc chleba máslem“, klidně položte na jeden z krajíců celou vaničku s máslem, neboť se nezmínili o tom, abyste vaničku nejdříve otevřeli.
    - Pokud instrukce říkají, abyste udělal něco, co nemůžete udělat, jako například „Použij nůž k nabrání džemu“, přestože je sklenice s džemem stále uzavřená, řekněte jim jednoduše ERROR (CHYBA) a nepokračujte dále.
- Jakmile projdete několika kroky nebo celými programy, většině žáků dojde, že musí své práce předělat, jelikož některé kroky vynechali nebo předpokládali, že jako robot už nějaké funkce umíte. Pravděpodobně budou chtít své papíry s pseudokódy nazpět, aby je mohli předělat.

Přepište pseudokód ve funkci:

- Dejte jim šanci k přepsání jejich „UdelejToast“ funkce a úkol zjednodušte tím, že jim řeknete, že několik malých základních funkcí umíte, aby je nemuseli psát. Například když napíšou „Otevři sklenici s džemem“ nebo „Sundej víčko ze sklenice s džemem“,

řekněte jim, že funkci „OtevriSklenici“ už umíte, aby nemuseli psát „Uchop pevně víčko sklenice. Otoč s víčkem doleva...“ atd.

- Žáci se rozhodně budou ptát „A umíte...?“, aby zjistili, jaké další funkce umíte. Zde si klidně můžete připravit výčet konkrétních funkcí, které už umíte.
- Předveďte pár z jejich předělaných funkcí. Jsou to většinou žáci, kteří rádi uvidí výsledek.

Příklad:

Hlavní programové funkce	Pomocné funkce
Udělej Toast • Otevři pytlík s chlebem • Odeber dva krajíce chleba • Polož každý krajíc jednou stranou dolů vedle sebe na talíř • OtevriMaslo • OtevriJam • Zvedni nůž • Namaz máslo na jeden krajíc chleba • Namaz džem na druhý krajíc chleba • Polož nůž dolů • Zvedni jeden krajíc chleba a polož ho namazanou stranou dolů na druhý krajíc chleba • Zabal chleba do papírového ubrousku	Otevři • Uchop pytlík na konci, na kterém se otvírá • Sundej plastové uzavírání • Rozevři pytlík • Sáhni dovnitř OtevriMaslo • Uchop jednou rukou vaničku s máslem • Druhou rukou zvedni víčko z vaničky • Polož víčko vrchní čistou stranu na stůl OtevriJam • Uchop jednu ruku okolo horního víčka sklenice a pevně ho stiskni • Druhou rukou uchop dolní část sklenice • Opakuj, dokud není víčko uvolněné, otoč horní rukou proti směru hodinových ručiček • Odstraň víčko ze sklenice Namaz • Sáhni nožem do sklenice • Naber obsah • Pohybuj nožem dopředu a dozadu po chlebu, dokud není nůž čistý • Opakuj předchozí 3 kroky, dokud není chléb zcela zakrytý

Úkol:

- Nechte každého žáka, aby si vybral jednoduchý úkol, jako je třeba čištění zubů či zavazování tkaničky u boty a pseudokódem napsal funkci k jejímu splnění.

- Spolu s pseudokódem by si měl každý žák na další hodinu přinést vše, co je potřeba k vykonání jeho funkce.
- Vyberte pseudokód a pomůcky některého ze žáků a nechte tento úkol předvést jeho spolužáka.
- Poté, co vás žáci viděli posledně předvádět úkol s toastem, budou plnit instrukce naprosto přesně podle vašeho minulého vzoru.

Tato cvičení pomáhají studentům uvědomit si hodnotu funkcí jako způsob organizování jejich programů a také to, jak může každá funkce obsahovat i příkazy dalších menších funkcí.

Dodatečná úloha: Uklízecí robot

- Jak byste naprogramovali robota, aby uklidil dům?
- Jsou zde nějaké úkoly, které jsou pro všechny místnosti v domě stejné? (vysávání, utírání prachu)
- Jsou zde nějaké úkoly, které jsou specifické pouze pro určité místnosti v domě? (uklidit toaletu, ustlat postel)
- Které úkoly můžete přiřadit k funkcím? (zvednoutVeci, utritPrach, zamest, vysat)
- Jak byste mohli rozdělit celkový úkol uklizení domu ve specifické funkce? (uklidKuchyn, uklidObyvaciPokoj, uklidKoupelnu atd.)

Zadejte žákům, aby pomocí pseudokódu napsali dva příklady pro uklizení místnosti v domě.

Aktivita: Osudový skok

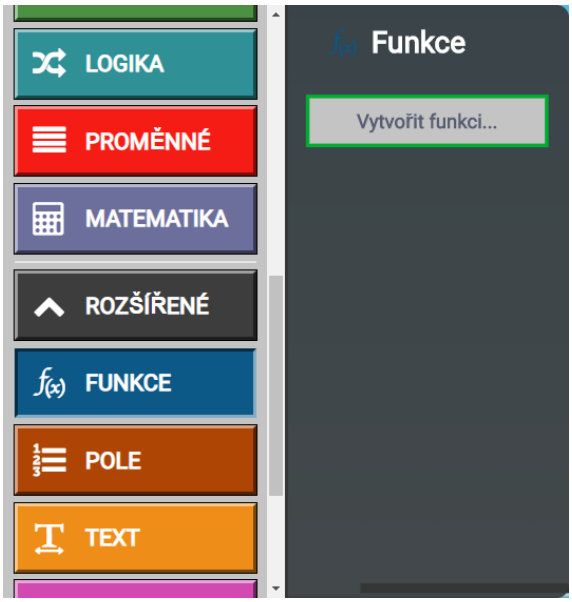
Hráči milují vytvářet v Minecraftu minihry pro své kamarády. V této aktivitě si zkusíme vytvořit jednoduchou minihru. Postavíme si malý bazének s vodou, vylezeme na malou plošinku 64 bloků nad zemí a jediná možnost, jak se dostat dolů, je skočit do bazénku s vodou. Pokud skočíte a vodu minete, zabijete se a prohrajete. Naštěstí budeme tuto minihru programovat v MakeCode pomocí funkcí, takže budeme moci zkoušet štěstí. Tento „osudový skok“ jste možná někdy viděli jako číslo v cirkuse.

Naše minihra se bude skládat ze 3 části:

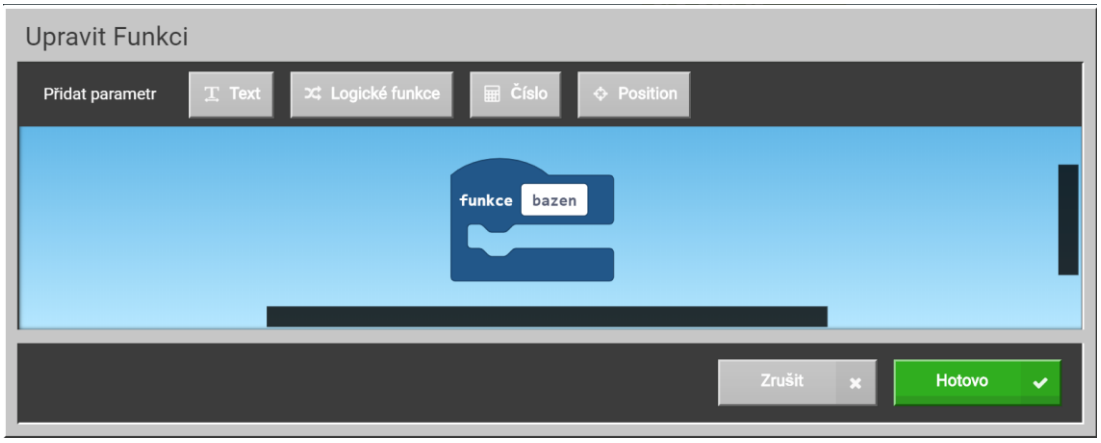
1. Vytvoření bazénku s vodou
2. Postavení plošiny
3. Přesunutí hráče na plošinu

Kroky:

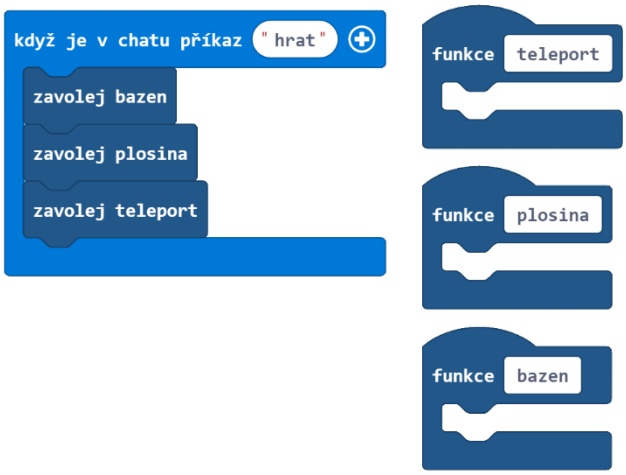
1. Vytvoř nový MakeCode projekt a pojmenuj ho „Skok“.
2. Klikni na ikonu Rozšířené k zobrazení dalších nabídek.
3. V nabídce Funkce klikni na „Vytvořit funkci“.



4. Pojmenuj si tuto funkci *bazen* a klikni na Hotovo. (Je možné vytvářet i funkce s parametry, parametry různých typů jednoduše přidáte při vytváření funkce.)



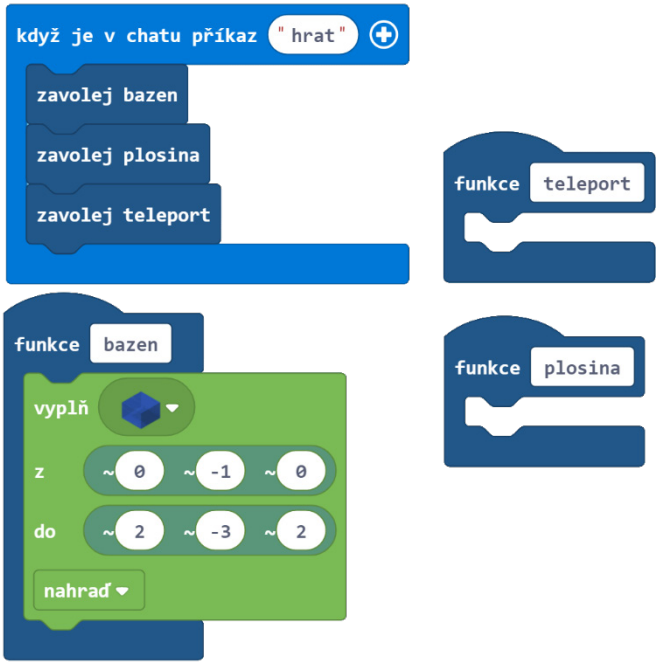
5. Zopakuj kroky 3 a 4, abyste si vytvořili další dvě funkce pojmenované: *plosina* a *teleport*.
6. Z nabídky Hráč přetáhni blok Když je v chatu příkaz.
7. Přejmenuj blok Když je v chatu příkaz na "hrat".
8. Z nabídky Funkce přetáhni do Když je v chatu příkaz "hrat" bloky: Zavolej bazen, Zavolej plosina, Zavolej teleport.



Tohle bude náš hlavní program k odstartování naší minihry. Nyní pojďte vytvořit tyto tři funkce.

Nejprve si vytvoříme bazének s vodou.

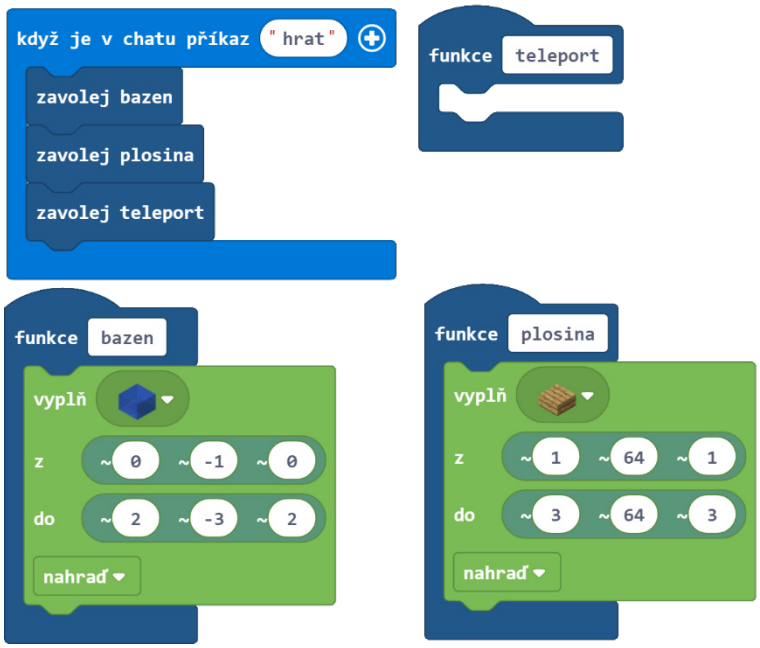
9. Z nabídky Bloky přetáhneme blok Vyplň a vložte ho do bloku Funkce bazen. Vyplň blok vyplní trojrozměrnou oblast od zadaných prvních souřadnic až po souřadnice druhé.
10. V bloku Vyplň vyber z rozbalovací nabídky blok vody.
11. Do bloku Vyplň napiš jako první souřadnici následující hodnoty: (0, -1, 0)
12. Do bloku Vyplň napiš jako druhé souřadnici následující hodnoty: (2, -3, 2)



Tento kód vytvoří bazén s vodou o velikosti 2 x 2 x 2, který se bude nacházet pod naším hráčem. Blok Vyplň je implicitně nastavený na *nahradit*. To znamená, že již existující bloky budou nahrazeny vodou.

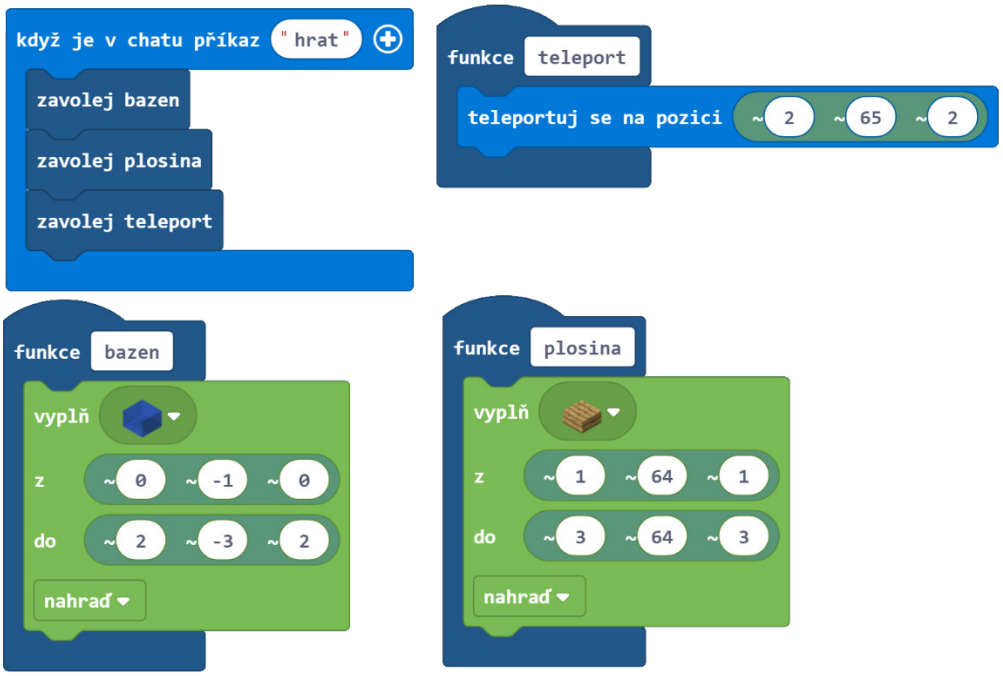
Jakmile bude bazén hotový, je zapotřebí postavit plošinu. Není k tomu potřeba nějaké věže nebo žebříku - je pouze jedna cesta dolů! Chceme vytvořit dřevěnou plošinu, která bude malinko posunutá oproti bazénu pod ní, aby měl hráč šanci skočit z plošiny přímo do bazénu.

13. Z nabídky Bloky přetáhni další blok Vyplň tentokrát do Funkce plosina.
14. V bloku Vyplň vyber z rozbalovací nabídky Panel ze sekvojového dřeva.
15. Do bloku Vyplň napiš jako první souřadnici následující hodnoty: (1, 64, 1).
16. Do bloku Vyplň napiš jako druhou souřadnici následující hodnoty: (3, 64, 3).



Posledním krokem je teleportace hráče do místa přesně nad střed platformy.

17. Z nabídky Hráč přetáhni Teleportuj se na pozici blok do Funkce teleport.
18. Do Teleportuj se na pozici bloku napiš následující souřadnice: (2, 65, 2), což jsou souřadnice plošiny.



Teď už můžeš vyzkoušet svoji minihru!

V Minecraft světě nastav herní režim na Přežití, najdi si otevřenou plochu a spusť svoji minihru napsáním příkazu „hrat“ do chatu. Poté zkus svůj osudový skok! Pamatuj, že když se pohybuješ po platformě, můžeš stisknutím klávesy shift předejít pádu dolů, zatímco budeš hledat ideální místo ke skoku. Nechte žáky, ať si vyzkouší minihry svých spolužáků.

Je to dlouhá cesta dolů!

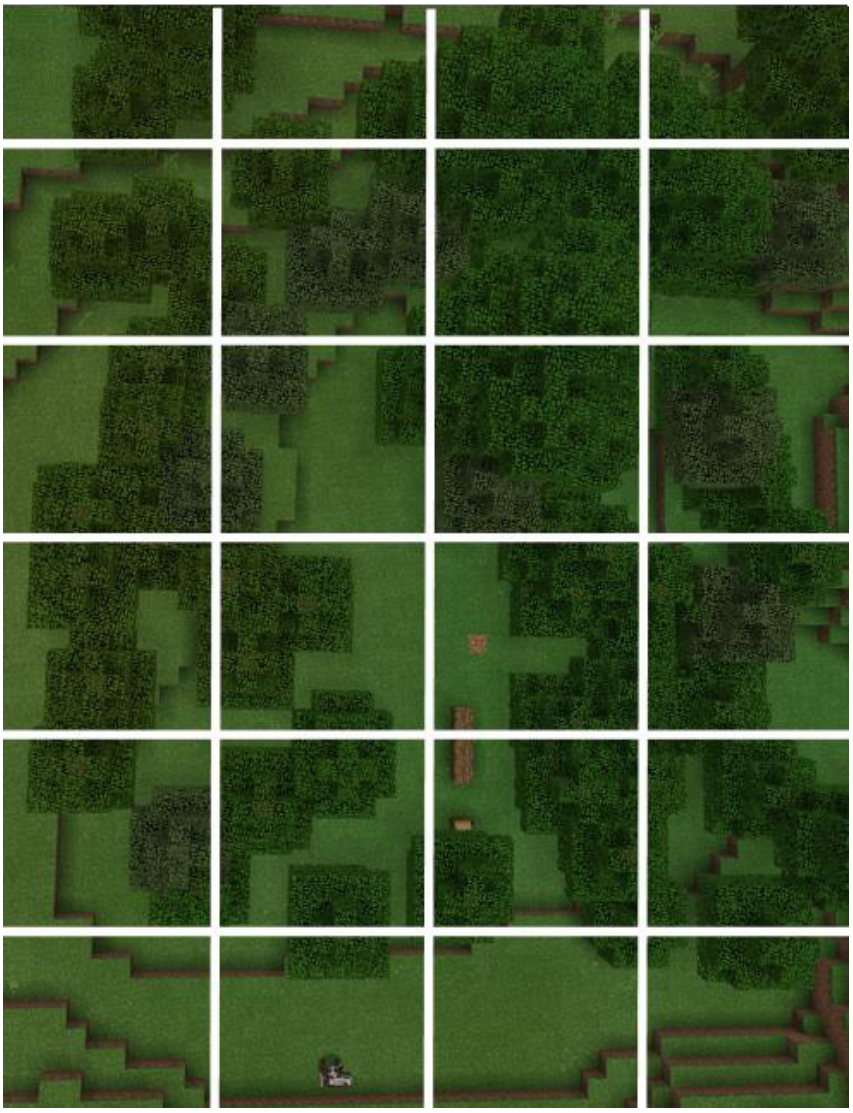
Aktivita: Inteligentní dřevorubec

Určitě si vzpomenete, jak jsme v lekci Podmínky programovali Agenta, aby uměl kácet stromy. Ke každému stromu však musel být pokaždé teleportován hráčem. Představte si, že

byste pustili inteligentnějšího Agenta volně po lese a nechali ho, ať si stromy najde sám a automaticky je pokácí bez potřeby zásahu jakéhokoliv hráče. Co vše by bylo zapotřebí?

Když vidíme les my, je pro nás jednoduché v něm rozpoznat stromy. U Agenta je to však jiné. Dokáže rozpoznat pouze bloky těsně před ním, nalevo a napravo od něj. Abychom dokázali naprogramovat inteligentního Agenta, představme si sami sebe prodírající se lesem se zavázanýma očima. Jak budete postupovat?

Jedna z možností, jak řešit tento problém, je rozdělit les mřížkou a procházet jím po řadách. Je to sice pomalé, ale nakonec prozkoumáme každý čtverec z mřížky. Efektivnější způsob může být s každým krokem otestovat čtverec před námi, nalevo a napravo.



Navíc potřebujeme vymyslet, jak naprogramujeme Agenta, aby se uměl pohybovat v kopcovitém terénu, jelikož stromy nerostou moc často na ploché planině. Dále potřebujeme zajistit, aby začal s kácením stromu vždy od jeho kořene.

A nakonec musím zapsat samotný program na těžbu dřeva. Můžeme použít ten, který jsme vytvořili v lekci Podmínky. Jelikož je to poměrně složitý program, rozdělíme si ho na různé části:

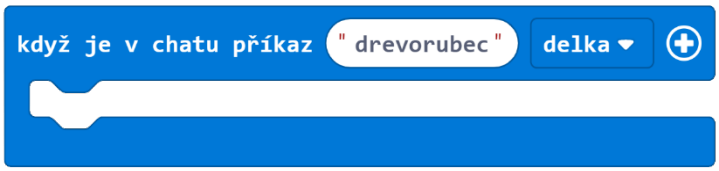
- Hlavní program drevorubec
- Hledání stromů (funkce vyhledat)
- Pohyb v terénu (funkce pohyb)
- Otočení se na konci řady (funkce otocit)
- Kácení stromů (funkce kacet)

Kroky:

1. Vytvoř v MakeCode nový program pojmenovaný „Inteligentní dřevorubec“.
2. Přejmenuj Když je v chatu příkaz blok na "drevorubec".

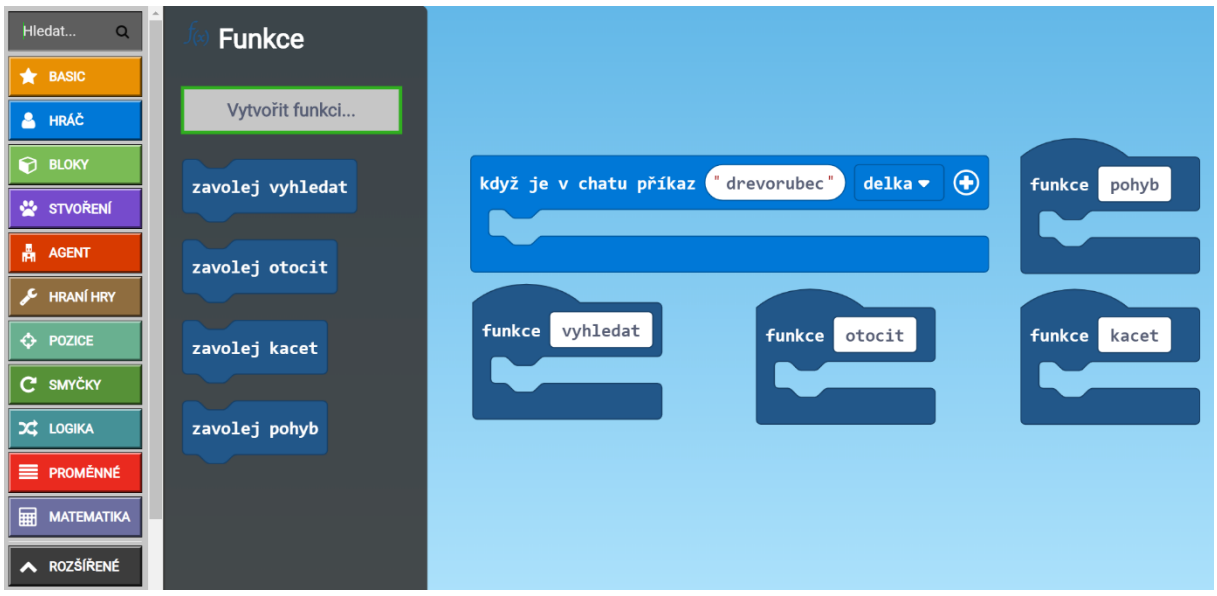
K určení velikosti prohledávané oblasti přidejme k našemu Drevorubec bloku parametr, který bude reprezentovat délku jedné strany čtverce oblasti. Například pokud zadáme „drevorubec 25“, Agent prohledá plochu o velikosti 25 x 25, která jistě bude plná stromů.

3. Klikni na znaménko plus (+) v Když je v chatu příkaz bloku a přidej parametr num1 a přejmenuj ho na *delka*.



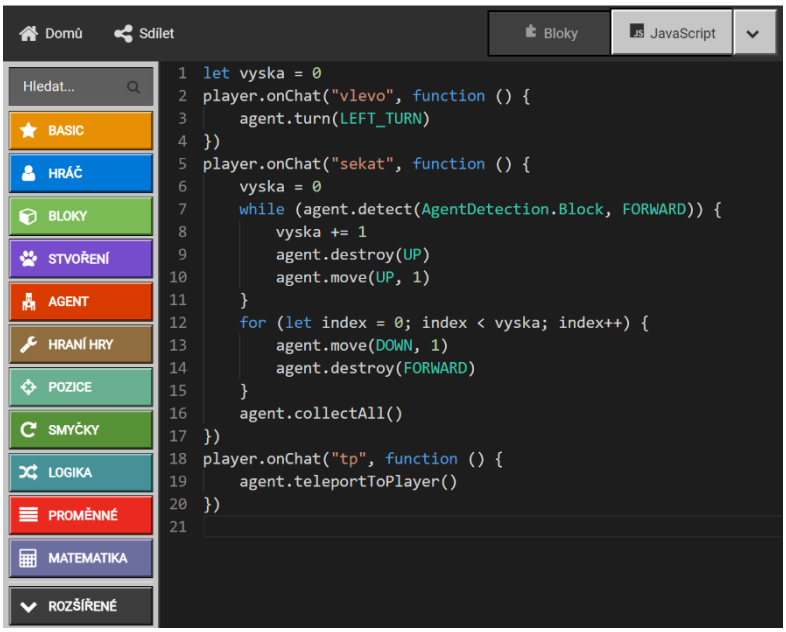
Nejdříve si vytvořme 4 funkce, které bude náš program obsahovat.

4. Klikni na nabídku Rozšířené na panelu nástrojů a zobrazí se další kategorie.
5. Po kliknutí na Funkce klikni čtyřikrát na tlačítko „Vytvořit funkci“.
6. Jednotlivé funkce pojmenuj: *vyhledat, pohyb, otocit* a *kacet.*

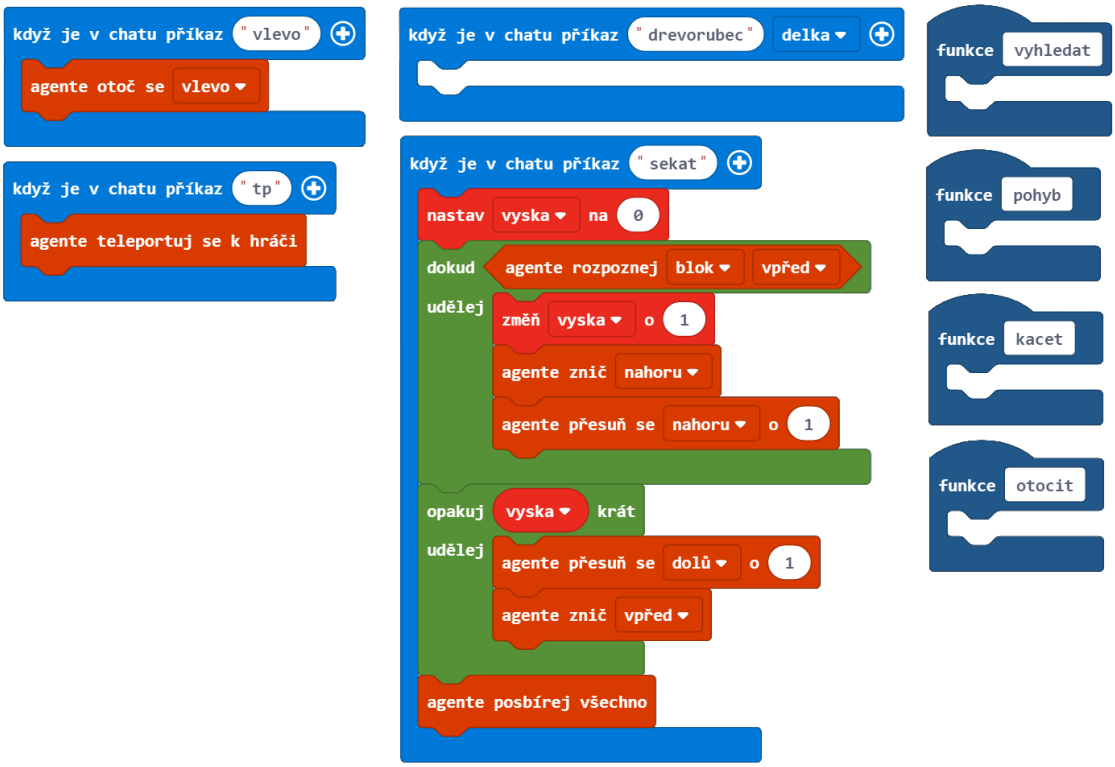


Začněme s funkcí kacet, kterou jsme si již vytvořili v lekci Podmínky.

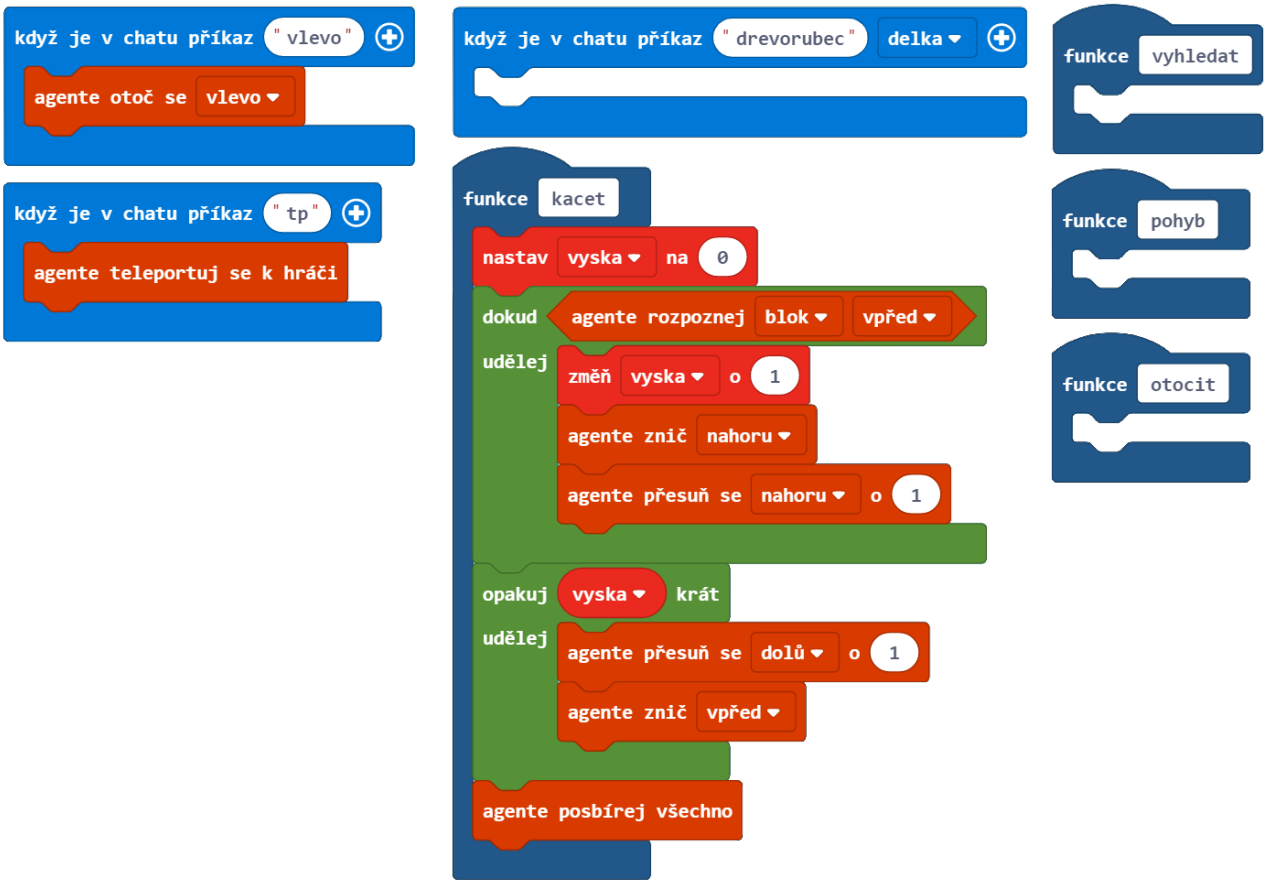
7. Otevři si projekt Dřevorubec, který jsme si v MakeCode vytvořili v předchozí kapitole.
8. Klikni na JavaScript tlačítko, přepni svůj program do JavaScriptu.
9. Označ si celý kód (Ctrl-A) a zkopíruj ho (Ctrl-C).



10. Klikni na tlačítko Domů, které najdeš v levém horním rohu. Přepneš se zpět na hlavní stránku MakeCode.
11. Otevři si opět projekt Inteligentní dřevorubec.
12. Klikni na JavaScript tlačítko a přepni se do JavaScript editoru.
13. Umísti kurzor své myši do posledního řádku a vlož tam zkopírovaný kód (Ctrl+V).
14. Klikni na tlačítko Bloky a vrať se zpět do Blokového editoru. Zkopírovaný kód by se zde měl objevit ve formě bloků.

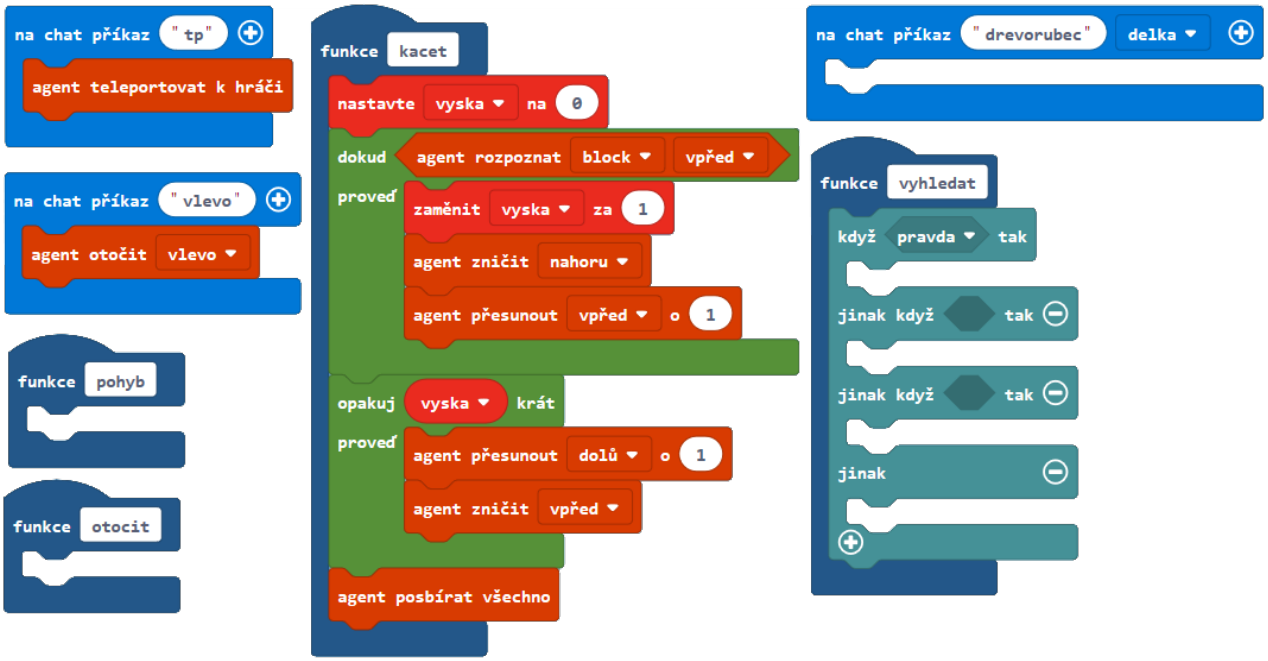


15. Přetáhni všechny bloky z příkazu Když je v chatu příkaz "sekat" do Funkce kacet.
16. Smaž Když je v chatu příkaz "sekat".

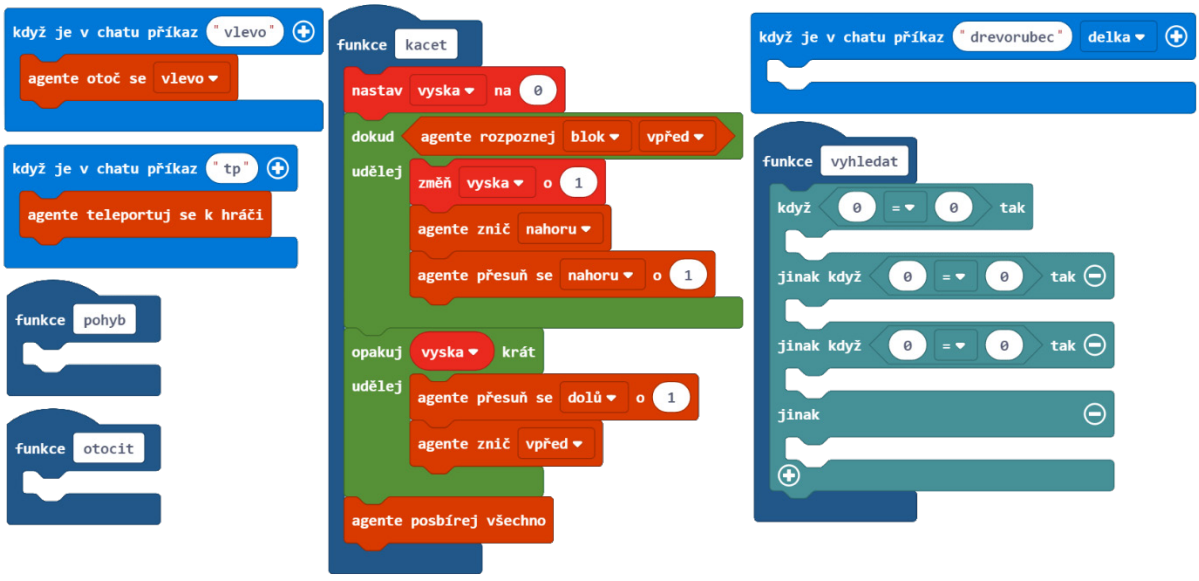


Nyní se vrhneme na funkci vyhledat, pomocí které Agent zjistí, zda se v okolí vyskytují nějaké stromy. Agent zkontroluje, zda se po jeho levici, pravici či před ním nenachází blok představující část stromu. Pokud najde blok dřeva, použije Funkci kacet a strom pokácí.

17. Z nabídky Logika přetáhni blok Když Tak Jinak do Funkce vyhledat.
18. Klikni dvakrát na znaménko plus (+) v Když Tak Jinak bloku, abys vytvořil další dva případy Jinak Když.

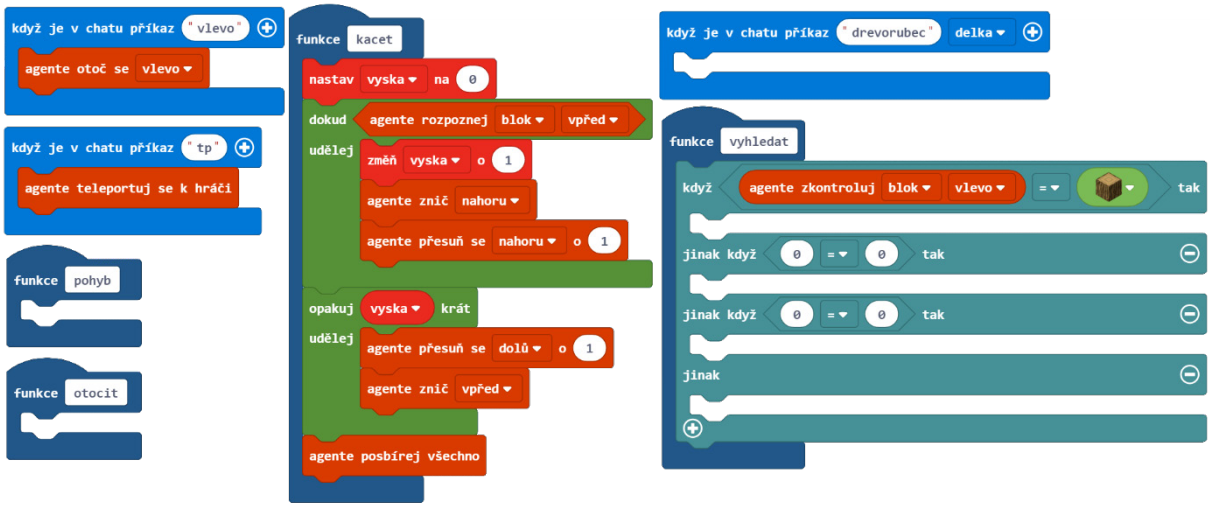


19. Z nabídky Logika přetáhni 3 „rovná se“ bloky do každého Když a Jinak Když slotu (Nebo můžete blok vložit jen jednou a pravým tlačítkem zvolit možnost Klonovat).



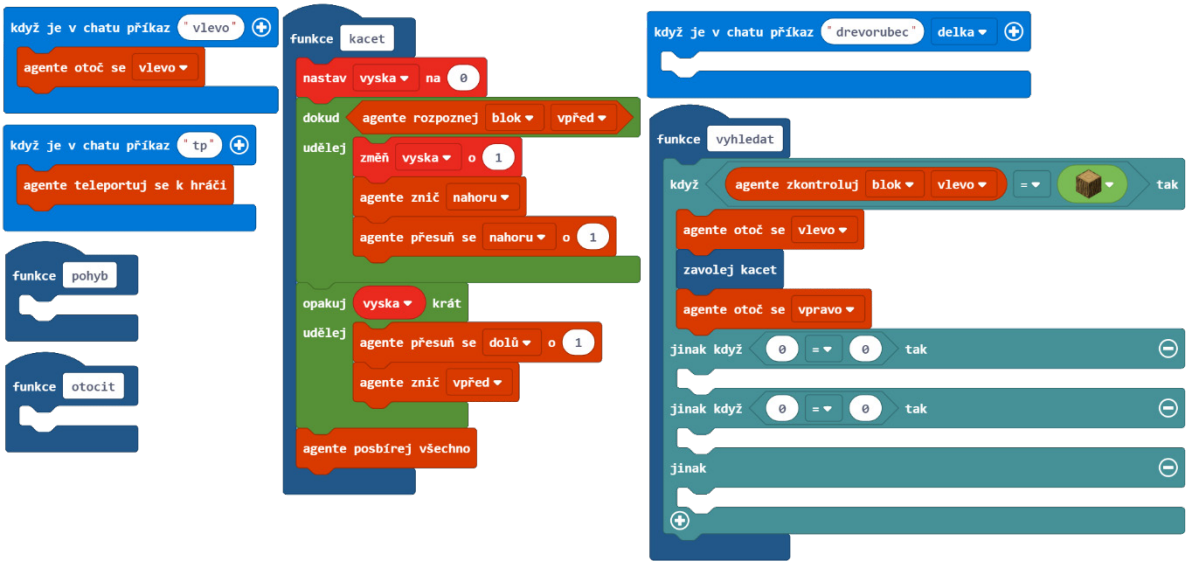
Nejprve pojďme zjistit, zda se nenachází blok dřeva nalevo od Agenta.

20. Z nabídky Agent přetáhni blok Agente zkontroluj do prvního slotu Rovná se do první podmínky Když.
21. V Agente zkontroluj bloku vyber z druhé rozbalovací nabídky směr „vlevo“.
22. Z nabídky Bloky přetáhni prvek Blok a vlož ho do druhého slotu Rovná se bloku.
23. V Blok prvku vyber z rozbalovací nabídky Dubové dřevo.



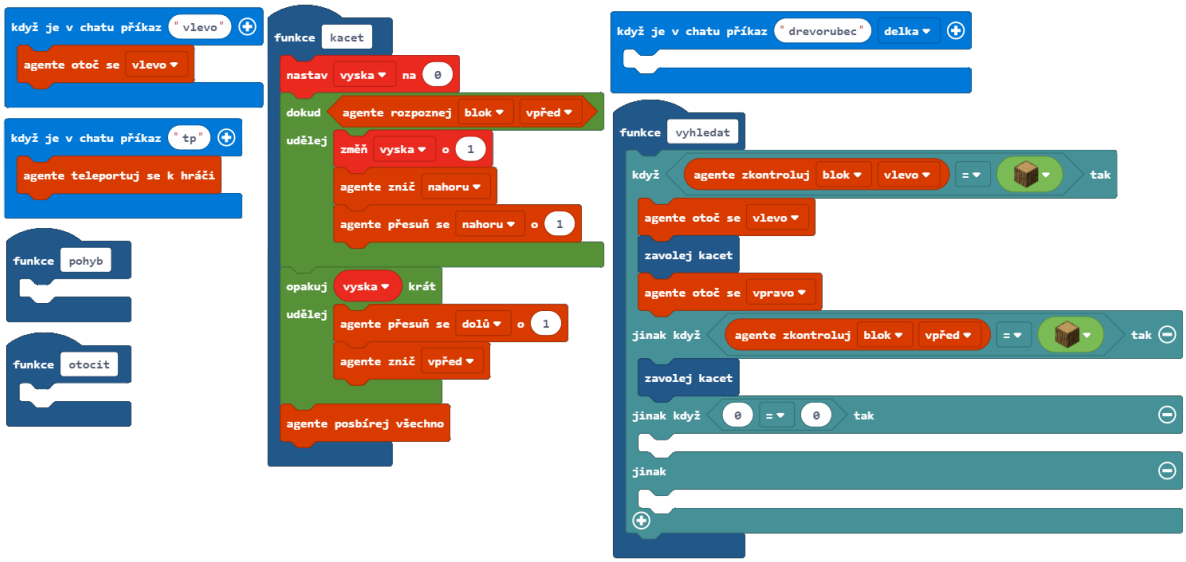
Pokud najdeme nalevo od Agenta dřevo, chceme, aby se Agent otočil doleva a provedl funkci kacet a strom pokácel. Poté musíme Agenta otočit zpět směrem dopředu.

24. Z nabídky Agent přetáhni blok Agente otoč se do podmínky Když Tak.
25. Z nabídky Funkce přetáhni Zavolej kacet pod Agente otoč se blok.
26. Z nabídky Agent přetáhni Agente otoč se blok pod Zavolej kacet blok.
27. V Agente otoč se bloku vyber z rozbalovací nabídky směr *"vpravo"*.



Nyní zkontrolujeme, zda se nenachází nějaké dřevo před Agentem. V tomto případě Agenta otáčet nepotřebujeme.

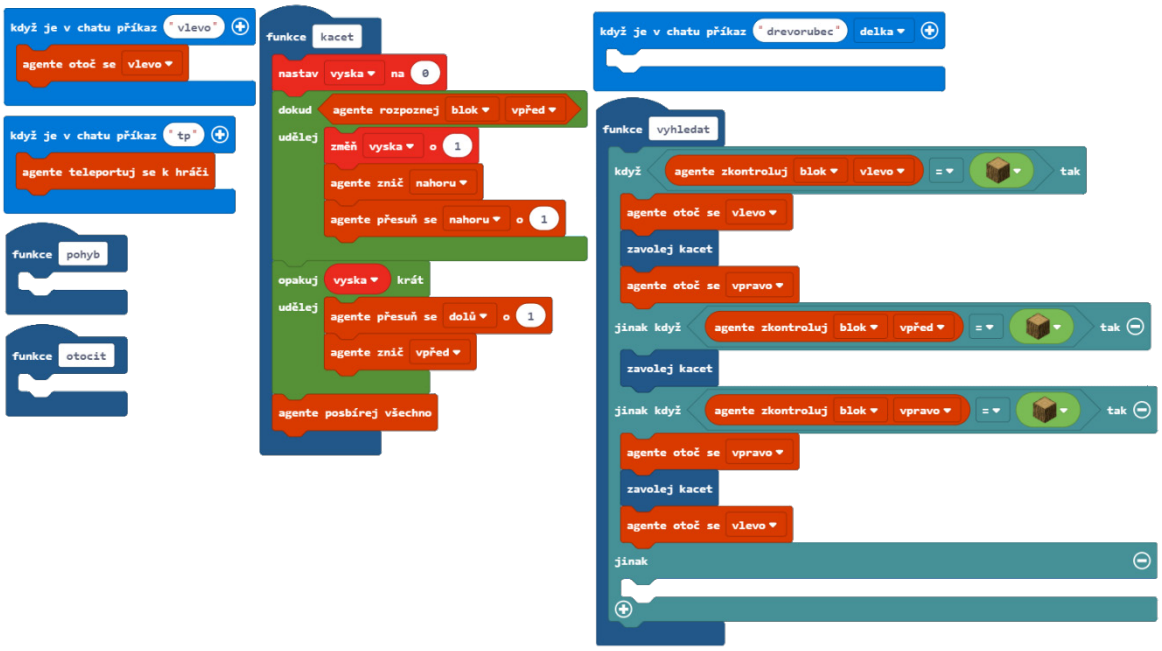
28. Z nabídky Agent přetáhni blok Agente zkontroluj do prvního slotu Rovná se bloku do první části podmínky Jinak Když.
29. Z nabídky Bloky přetáhni prvek Blok a vlož ho do druhého slotu Rovná se bloku.
30. V prvku Blok vyber z rozbalovací nabídky Dubové dřevo.
31. Z nabídky Funkce přetáhni Zavolej kacet blok pod první část podmínky Jinak Když.



A nakonec se podívejme, zda se nenachází nějaké dřevo napravo od Agenta.

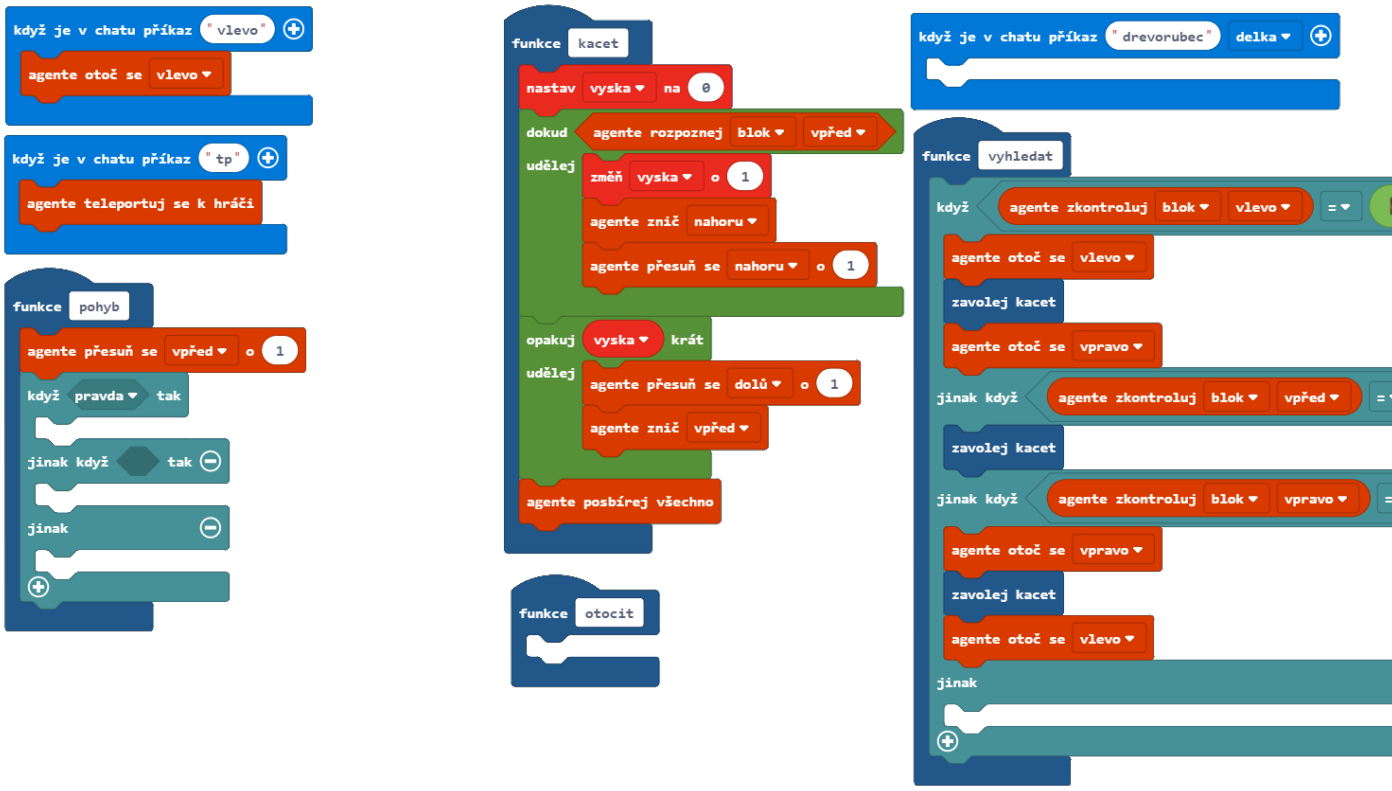
32. Z nabídky Agent přetáhni Agente zkontroluj blok do prvního slotu Rovná se bloku do druhé podmínky Jinak Když.
33. V Agente zkontroluj bloku vyber z rozbalovací nabídky směr „vpravo“.
34. Z nabídky Bloky přetáhni prvek Blok a vlož ho do druhého slotu Rovná se bloku.
35. V prvku Blok vyber z rozbalovací nabídky Dubové dřevo.
36. Z nabídky Agent přetáhni Agente otoč se blok a vlož ho do podmínky Jinak Když.
37. V Agente otoč se bloku vyber z rozbalovací nabídky směr „vpravo“.
38. Z nabídky Funkce přetáhni Zavolej kacet blok pod Agente otoč se blok.
39. Z nabídky Agent přetáhni Agente otoč se blok pod Zavolej kacet blok.

Pokud se už nalevo, napravo a ani před agentem nevyskytuje žádné další dřevo, Agent neprovede žádnou akci, a tudíž poslední Jinak klauzule zůstane prázdná.

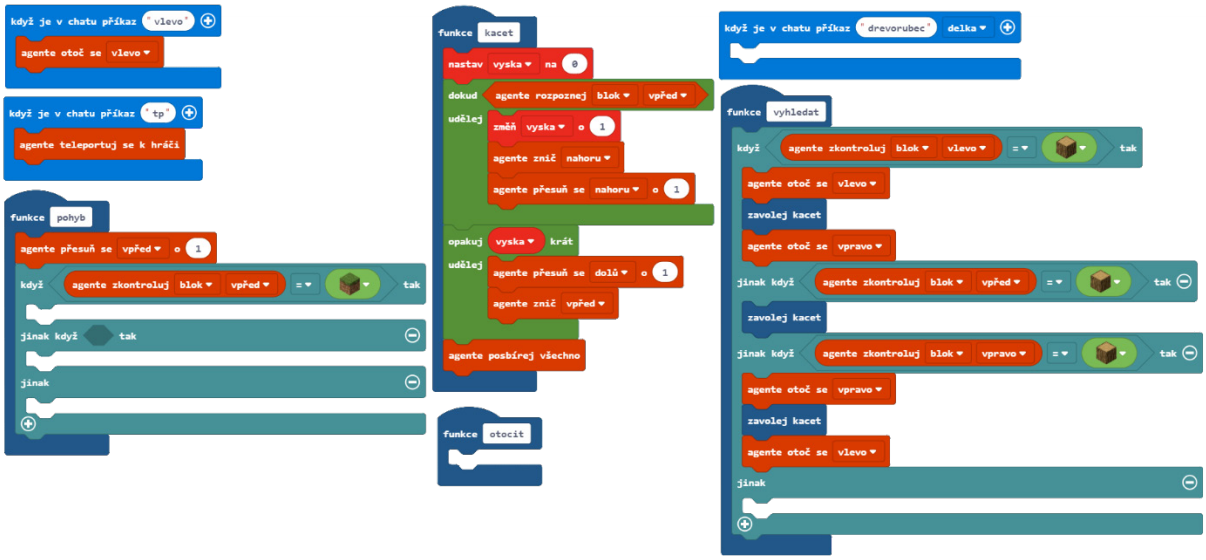


Nyní se pojďme soustředit na funkci Pohyb, abychom naučili Agenta pohybovat se na základě nerovností terénu. Abychom přiměli Agenta hledat kořeny stromů, musíme s ním pohybovat nahoru vždy, když před ním bude nějaký blok, a dolů, když pod ním naopak žádný blok nebude (Agent bude kopírovat terén).

40. Z nabídky Agent přetáhni Agente přesuň se blok do Funkce pohyb.
41. Z nabídky Logika přetáhni Když Tak Jinak blok pod Agente přesuň se blok.
42. V Když Tak Jinak bloku klikni na znaménko plus (+), abys vytvořil podmínku Jinak Když.

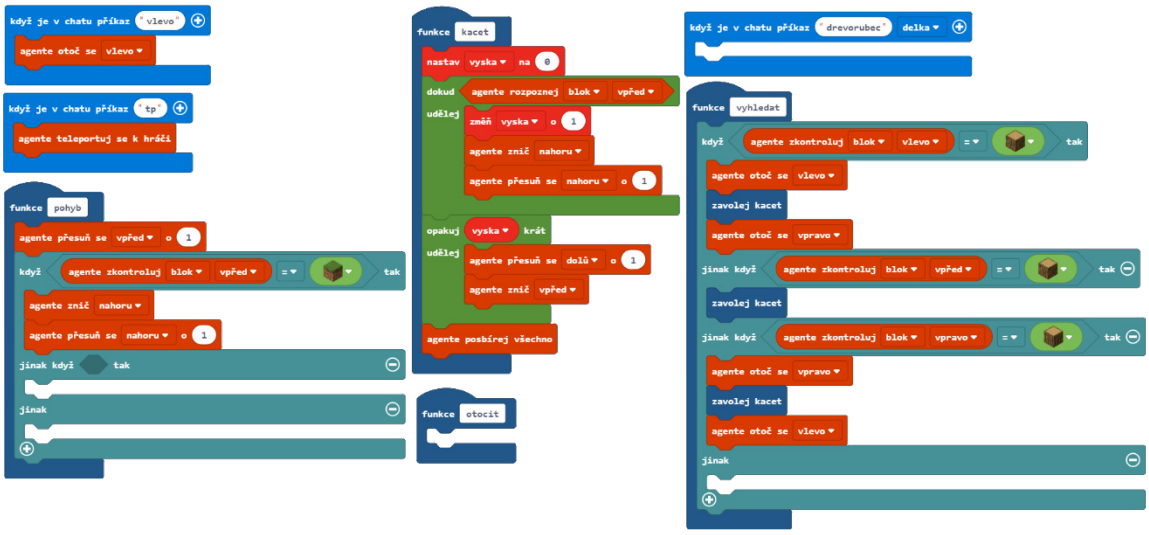


43. Z nabídky Logika přetáhni Rovná se (=) blok do podmínky Když a nahraď jím „pravda“.
44. Z nabídky Agent přetáhni Agente zkontroluj blok do prvního slotu Rovná se bloku.
45. Z nabídky Bloky přetáhni prvek Blok do druhého slotu Rovná se bloku.



Pokud je blok nacházející se před Agentem blok trávy, musíme Agenta zvednout o blok vzhůru. Musíme se ale také ujistit, že Agenta neblokuje nic nad ním (například listy).

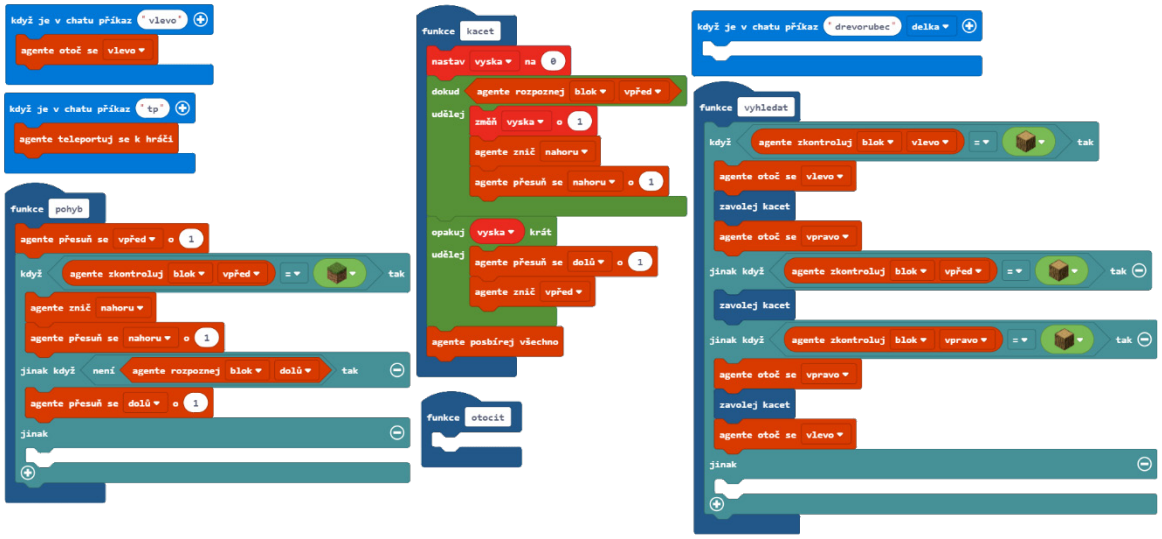
46. Z nabídky Agent přetáhni Agente znič blok do podmínky Když Tak.
47. V Agente znič bloku vyber z rozbalovací nabídky směr "nahoru".
48. Z nabídky Agent přetáhni Agente přesuň se blok pod Agente znič blok
49. V Agente přesuň se bloku vyber z rozbalovací nabídky směr "nahoru"



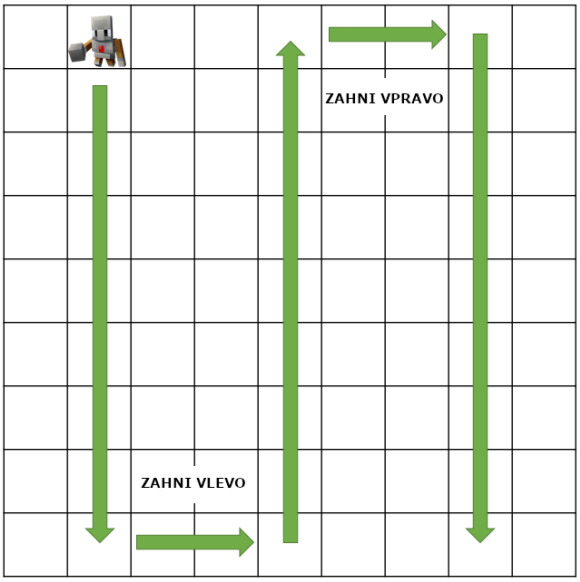
Teď se postaráme o posunutí Agenta o blok níže vždy, když se pod ním nebude nacházet blok.

50. Z nabídky Logika přetáhni Není blok do podmínky Jinak Když.
51. Z nabídky Agent přetáhni Agente rozpoznej blok do Není bloku.
52. V Agente rozpoznej bloku vyber z druhé rozbalovací nabídky směr „dolů“. To v podstatě znamená, že podmínka je splněna, když Agent pod sebou nezaznamenal blok.
53. Z nabídky Agent přetáhni Agente přesuň se blok pod podmínku Jinak Když.
54. V Agente přesuň se bloku vyber z rozbalovací nabídky směr „dolů“.

Funkce Pohyb bude s Agentem pohybovat vpřed a kontrolovat, zda se před ním nenachází blok hlíny. Pokud ANO, posune se Agent nahoru. Pokud před ním blok nebude, zkontroluje, zda je blok pod ním. Pokud NE, posune se Agent dolů. Pokud před ním ani pod ním není blok, neudělá Agent nic.



V poslední funkci otočit je potřeba, aby se Agent na konci každé řady otočil. Musí se ale otáčet střídavě doleva a doprava.



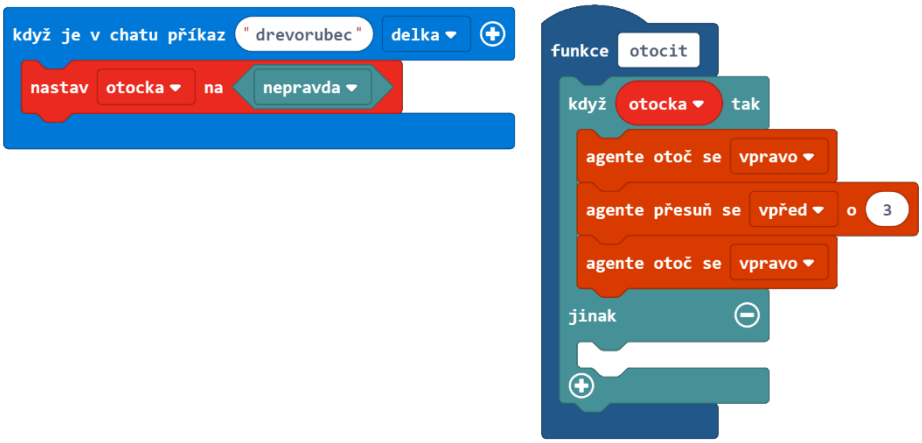
Jaká je nejlepší možnost, jak zaručit, aby se Agent otáčel střídavě?

Vytvořme booleovu proměnnou, abychom si zajistili správný směr otáčení Agenta. Vzpomeňte si na hodiny Proměnných. Booleova proměnná má pouze dvě možné hodnoty: Pravda a Nepravda. V našem případě použijeme Pravda pro otočení se doprava a Nepravda pro otočení se doleva. Jakmile se Agent otočí, změníme hodnotu proměnné na její opačnou hodnotu použitím bloku Není.

55. V nabídce Proměnné klikni na tlačítko „Vytvořit proměnnou“.
56. Tuto proměnou pojmenujeme „otocka“.

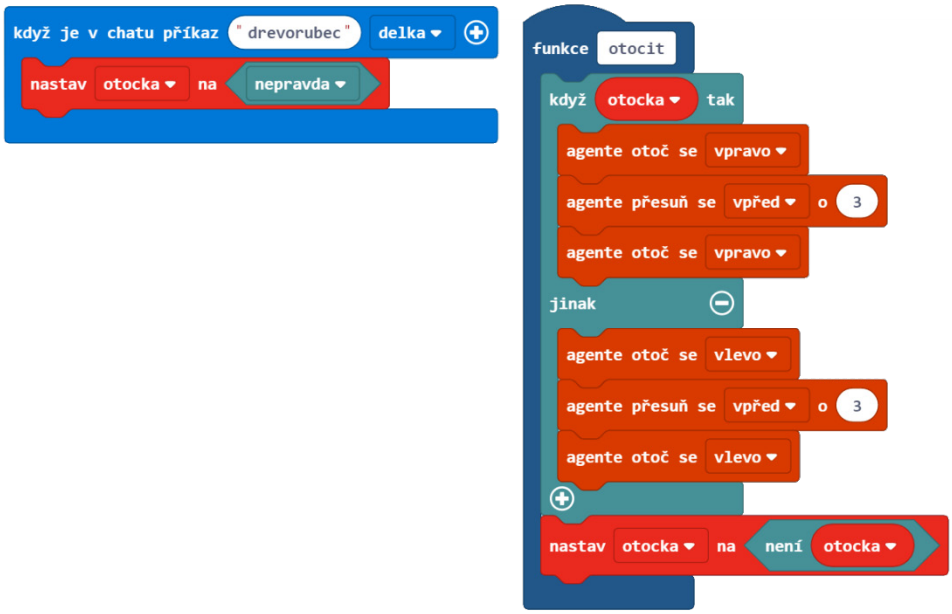
57. Z nabídky Proměnné přetáhni blok Nastav do bloku Když je v chatu příkaz "drevorubec".
58. V bloku Nastav vyber z rozbalovací nabídky proměnnou „otocka“.
59. Z nabídky Logika přetáhni blok Nepravda do bloku Nastav a nahraď jím hodnotu 0.

60. Z nabídky Logika přetáhni Když Tak Jinak blok do bloku Funkce otocit.
61. Z nabídky Proměnné přetáhni proměnnou otocka a vlož ji do podmínky Když místo „pravda“.
62. Z nabídky Agent přetáhni blok Agente otoč se do podmínky Když Tak.
63. V Agente otoč se bloku vyber z rozbalovací nabídky směr „vpravo“.
64. Z nabídky Agent přetáhni blok Agente přesuň se pod blok Agente otoč se.
65. Napiš 3 do slotu bloku Agente přesuň se, aby se Agent posunul o 3 bloky dopředu.
66. Z nabídky Agent přetáhni další blok Agente otoč se pod blok Agente přesuň se.
67. V Agente otoč se bloku vyber z rozbalovací nabídky směr „vpravo“.



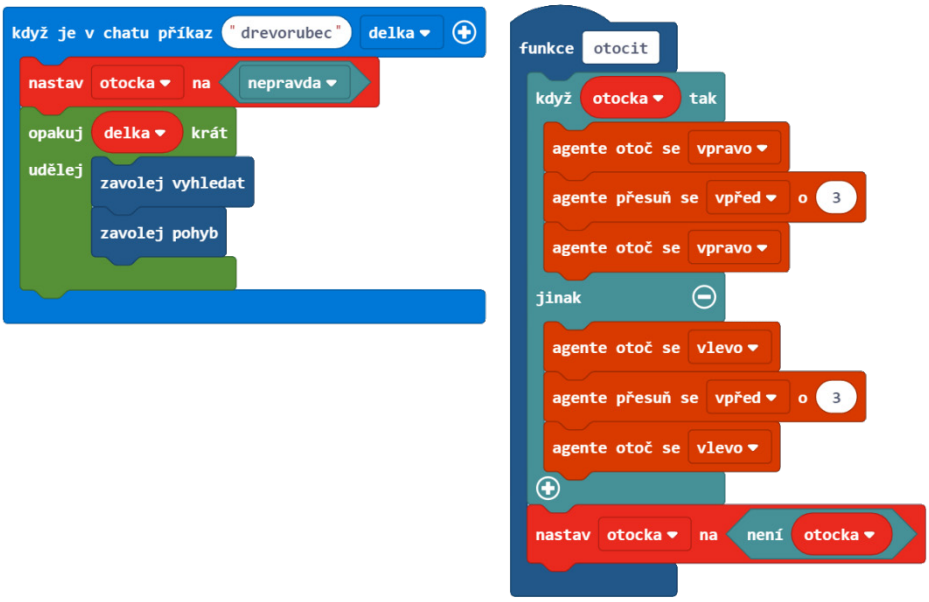
68. Z nabídky Agent přetáhni blok Agente otoč se do podmínky Jinak.
69. Z nabídky Agent přetáhni blok Agente přesuň se pod blok Agente otoč se.
70. Do slotu bloku Agente přesuň se napiš 3, aby se Agent posunul o 3 bloky dopředu.
71. Z nabídky Agent přetáhni blok Agente otoč se pod blok Agente přesuň se.
72. Nakonec si z nabídky Proměnné přetáhni blok Nastav a vlož ho za podmínku Když Tak Jinak do Funkce otocit
73. V Nastav bloku vyber z rozbalovací nabídky proměnnou otocka.
74. Z nabídky Logika přetáhni blok Není do slotu bloku Nastav namísto hodnoty 0.
75. Z nabídky Proměnné přetáhni proměnnou otocka do bloku Není.

Pokud je hodnota proměnné „otocka“ Pravda, Agent se otočí o 90° doprava, posune se o 3 bloky dopředu a ještě jednou se otočí o 90° doprava. Pokud je hodnota proměnné „otocka“ Nepravda, Agent se otočí o 90° doleva, poté se posune o 3 bloky vpřed a ještě jednou se otočí o 90° doleva. Poté nastavíme hodnotu proměnné „otocka“ na její opačnou hodnotu.



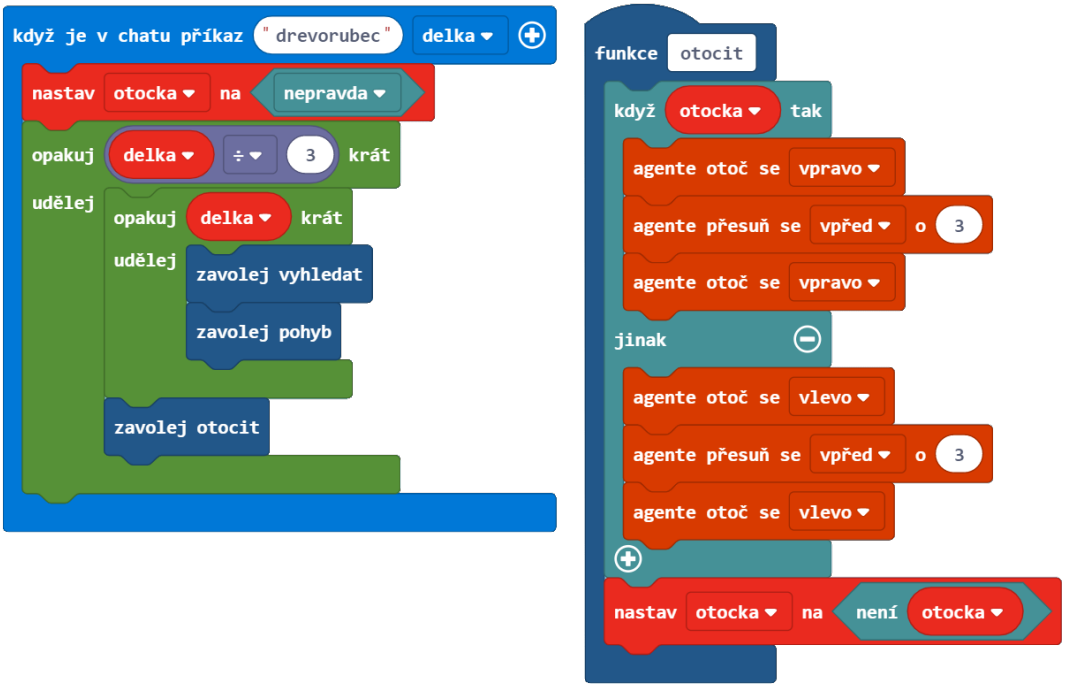
Nyní když už máme všechny potřebné funkce vytvořené, sestavíme je dohromady. V každém kroku chceme, aby Agent hledal stromy a pohyboval se společně s terénem. Pamatujte, že jsme si nastavili parametr „delka“ jako počet bloků v každé řadě.

76. Z nabídky Smyčky přetáhni blok Opakuj do Když je v chatu příkaz "drevorubec" pod blok Nastav.
77. Z nabídky Proměnné přetáhni proměnnou delka do bloku Opakuj a nahraď jím číslo 4.
78. Z nabídky Funkce přetáhni bloky Zavolej vyhledat a Zavolej pohyb do cyklu Opakuj.



Nyní chceme, aby se náš Agent na konci každé řady otočil. Představme si situaci, že jsme Agentovi dali příkaz „vyhledat 25“, což by znamenalo, že prohledá oblast 25 x 25 bloků. Každým krokem ale kontrolujeme tři bloky najednou. Můžeme tedy 25 vydělit 3 a snížit počet kroků k prozkoumání celé oblasti. Pokud tuto podmínku budeme chtít zahrnout v proměnné delka, musíme delka vydělit 3. Výsledek vložíme do vnějšího cyklu.

79. Z nabídky Smyčky přetáhni další cyklus Opakuj a vlož ho okolo již existujícího cyklu Opakuj v Když je v chatu příkaz "drevorubec". Ujisti se, že blok Nastav je stále mimo Opakuj cyklus.
80. Z nabídky Matematika přetáhni blok Dělení (÷) do cyklu Opakuj místo čísla 4.
81. Z nabídky Proměnné přetáhni proměnnou delka do prvního slotu bloku Dělení místo čísla 0.
82. Do druhého slotu v bloku Dělení napiš číslo 3.
83. Z nabídky Funkce přetáhni blok Zavolej otocit do vnějšího cyklu Opakuj, ale za vnitřní Opakuj cyklus.

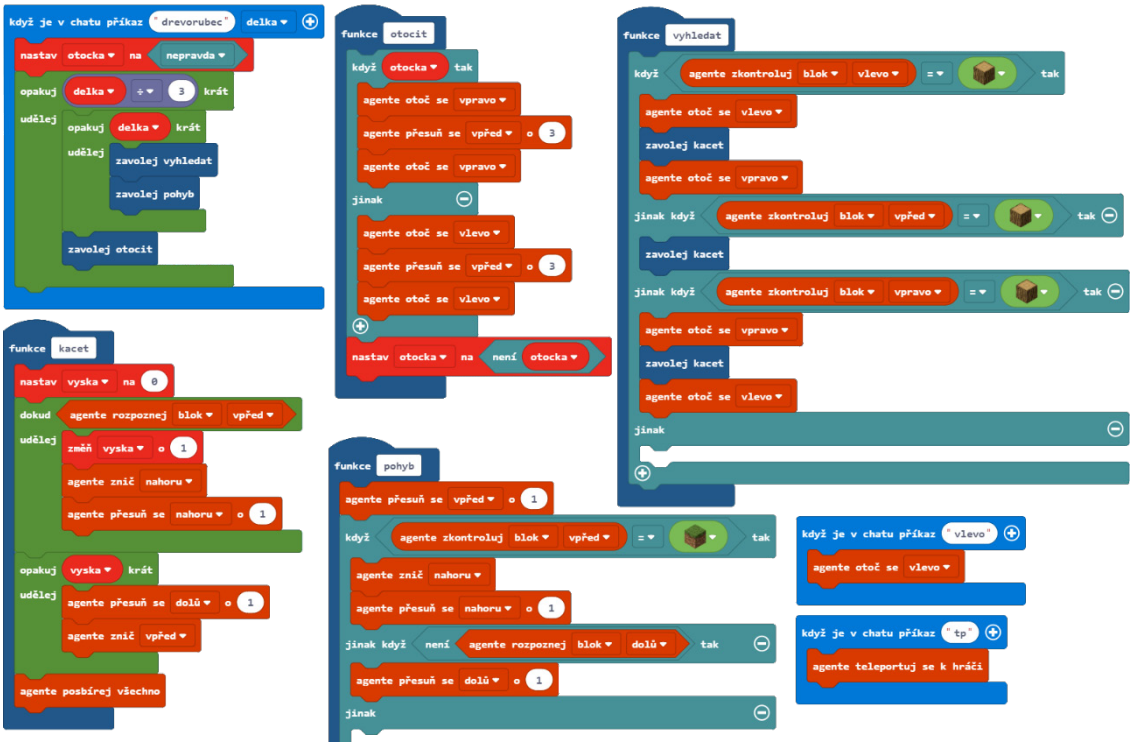


Volitelná rozšíření

Uvědomte si, že funkce Vyhledat bude fungovat pouze na travnatých blocích a pouze na terénu, který se zvyšuje či snižuje vždy pouze o jeden blok.

- Upravte kód tak, aby dokázal zpracovat stromy o rozměrech 2 x 2. Takovéto obří stromy naleznete například v džunglích nebo jako tmavé dubové stromy.
- Upravte kód tak, aby dokázal zpracovat akáciové stromy, které mají diagonální tvar.
- Upravte kód tak, aby dokázal pracovat s větvemi stromů.
- Pokud se okolo Agenta nachází více druhů stromů, Agent pokácí pouze jeden ze stromů – v pořadí zleva doprava. Najdi způsob, jak naučit Agenta pokácet všechny stromy okolo něj.

Kompletní program:



JavaScript:

```
let otocka = false
let vyska = 0
player.onChat("drevorubec", function (delka) {
    otocka = false
    for (let index = 0; index < delka / 3; index++) {
        for (let index = 0; index < delka; index++) {
            vyhledat()
            pohyb()
        }
        otocit()
    }
})
function pohyb () {
    agent.move(FORWARD, 1)
    if (agent.inspect(AgentInspection.Block, FORWARD) ==
 GRASS) {
        agent.destroy(UP)
        agent.move(UP, 1)
```

```
    } else if (!(agent.detect(AgentDetection.Block, DOWN
))) {
        agent.move(DOWN, 1)
    } else {

    }
}
player.onChat("vlevo", function () {
    agent.turn(LEFT_TURN)
})
function vyhledat () {
    if (agent.inspect(AgentInspection.Block, LEFT) == LO
G_OAK) {
        agent.turn(LEFT_TURN)
        kacet()
        agent.turn(RIGHT_TURN)
    } else if (agent.inspect(AgentInspection.Block, FORW
ARD) == LOG_OAK) {
        kacet()
    } else if (agent.inspect(AgentInspection.Block, RIGH
T) == LOG_OAK) {
        agent.turn(RIGHT_TURN)
        kacet()
        agent.turn(LEFT_TURN)
    } else {

    }
}
function otocit () {
    if (otocka) {
        agent.turn(RIGHT_TURN)
        agent.move(FORWARD, 3)
        agent.turn(RIGHT_TURN)
    } else {
        agent.turn(LEFT_TURN)
```

```
        agent.move(FORWARD, 3)
        agent.turn(LEFT_TURN)
    }
    otocka = !(otocka)
}
player.onChat("tp", function () {
    agent.teleportToPlayer()
})
function kacet () {
    vyska = 0
    while (agent.detect(AgentDetection.Block, FORWARD))
{
        vyska += 1
        agent.destroy(UP)
        agent.move(UP, 1)
    }
    for (let index = 0; index < vyska; index++) {
        agent.move(DOWN, 1)
        agent.destroy(FORWARD)
    }
    agent.collectAll()
}
```

Sdílený projekt: https://makecode.com/_e0h6sHUx1XWj

Samostatný projekt: Dům snů

V této lekci jsme se naučili, jak si uspořádat bloky kódu do oddělených funkcí. Tím ušetříme místo na pracovní ploše a kódy se stanou čitelnější. Funkce obvykle obsahují části kódu, které spolu logicky souvisejí a vykonávají jednu věc. Jméno funkce nám většinou prozradí, co daná funkce dělá (obvykle to bývá sloveso). Funkce v MakeCode mohou mít i několik typů parametrů. Parametr můžeme použít i v bloku Když je v chatu příkaz, uložit ho do již existující proměnné, ke které může mít přístup jakákoliv funkce.

Nyní je už jen na tobě, aby si vyzkoušel funkce při stavění v Minecraftu. První věcí, kterou každý hráč v Minecraftu udělá, je to, že si postaví dům. Svůj dům si můžeš postavit z různých druhů dřeva, postavit velká okna a využít tak přirozené denní světlo, nebo naopak žít v tmavém domě plném pastí na nevítané hosty. Může mít vysoké věže podobné těm na zámcích, nebo být postavený několik podlaží pod povrchem země a na povrchu být zamaskovaný do přirozeného prostředí. Také můžeš plochu okolo sebe použít k pěstování

mrkví a obilí, abys měl ty a tvá zvířata co jíst. Stavění je pro tebe šance ukázat a rozvinout tvoji kreativitu. MakeCode ti pomůže naprogramovat Agenta, aby ti se stavěním pomohl. Můžeš tak ostatním ukázat, jak ses zdokonalil v programování!

V tomto samostatném projektu si postav svůj vysněný domov nebo repliku již existující budovy. Ať si vybereš cokoliv, použij ke stavění funkce Zautomatizuj co nejvíce částí budovy. Pravděpodobně bys měl začít vytvořením příkazu, který vytvoří čtvercovou podlahu. Na podlahu budeš možná chtít položit koberec. To by Agent mohl také zvládnout. A co takhle Stavitel, který za tebe vytvoří farmu?

Vymysli originální funkce, z nichž každá bude splňovat tyto podmínky:

- Vytvoří, nebo alespoň pomůže vytvořit, budovu, pevnost, monument či jiný kousek architektury v Minecraftu.
- Vytvoř alespoň tři oddělené a odlišné funkce.
- Zvol vhodné názvy, které budou popisovat, co funkce dělá.

Pár příkladů možných funkcí:

- Polož koberec.
- Postav zdi místností.
- Postav fontánu.
- Postav zámeckou věž.
- Srovnej terén, aby budova stála na rovině.
- Postav plavecký bazén.

Minecraft deník

Zapište si do deníku:

- Co ses rozhodl postavit a proč?
- Popiš jednotlivě každou z funkcí. Co dělá?
- Jaké budovy ses rozhodl postavit sám? Proč?
- Vlož alespoň jeden snímek obrazovky své hotové práce.
- Sdílej svůj projekt na webu a nezapomeň pak vložit URL odkaz.

Hodnocení

	1	2	3	4
Deník	V deníku chybí odpovědi na 4 nebo více otázek.	V deníku chybí odpovědi na 2 nebo 3 otázky	V deníku chybí odpovědi na 1 otázku.	Deník obsahuje všechny odpovědi.
Projekt	Projekt je nesmyslný anebo neefektivní.	Projektu chybí 2 z požadovaných částí nebo je nesmyslný anebo neefektivní.	Projektu chybí 1 z požadovaných částí a je poměrně efektivní.	Projekt obsahuje všechny požadované části, je smysluplný a efektivní.

Funkce	Žádná z funkcí není logicky oddělena ani správně pojmenována.	Jedna funkce je logicky oddělena a správně pojmenována.	Dvě funkce jsou logicky odděleny a správně pojmenovány.	Alespoň tři funkce jsou logicky odděleny a správně pojmenovány.

Pole

Co jsou to pole?

V informatice je pole série míst, kam můžeme ukládat věci. Můžeme si ho představit třeba jako seznam položek, sérii poštovních schránek nebo nákladní vlak.

Naučili jsme se, že proměnné se používají k ukládání informací. Proměnné pomáhají počítači najít a vyjmout informaci uloženou někde ve své paměti. Můžeme proměnné pojmenovat, takže když o nich mluvíme jménem, počítač si proměnnou podle jména najde a zjistí, kde v paměti je daná informace uložená, kolik místa se očekává, že zabere, a o jaký typ informace jde.

Když máte hodně proměnných, se kterými pracujete, nebo pokud předem nevíte, kolik proměnných budete potřebovat, je šikovnější použít pole (array). Pole je způsob organizace několika proměnných v sérii, takže můžete jít rovnou popořadě a získat informace z každé z nich.

Lokace dané položky v poli se označuje jako její *index*. První položka v poli má vždy index 0. Následující má index 1 atd.

Délka (velikost) pole je stejná jako množství položek v poli. Délka je vždy o jednu větší než index poslední položky (protože indexy začínají na 0).

Na obrázku je 7 schránek s indexy 0 až 6.

Pole prasat v Minecraftu – všimněte si, že index čtvrtého prasete je 3, protože indexy začínají nulou.

V MakeCode můžete do polí ukládat různé typy věcí:

- Čísla
- Text
- Zvířata
- Pozice
- Bloky
- Příšery

Kdykoli můžete objekty přidávat, odstraňovat a počítat množství věcí v poli. Dokonce můžete pomocí jediného příkazu otočit jejich pořadí v poli. Pole jsou velice vhodný a mocný způsob práce s mnoha položkami najednou.

Třídění hodnot polí

Jakmile začnete do pole ukládat různé hodnoty, pravděpodobně je budete chtít i roztřídit. V informatice existují jisté druhy strategií nebo algoritmů k třídění hodnot. A schopnost porozumět, jak tyto různé třídící algoritmy fungují, je důležitá část informatiky. Jak budou žáci pokračovat ve studiu, naučí se další algoritmy, stejně jako jejich relativní účinnost. Některé z těchto algoritmů:

- Selection sort (výběrové třídění)
- Insertion sort (vkládací třídění)
- Quick sort (rychlé třídění)
- Merge sort (slučovací třídění)
- Bubble sort (bublinové třídění)
- Shell sort (schránkové třídění)

Zde je hezké video, které názorně ukazuje fungování některých třídících algoritmů polí: https://www.youtube.com/watch?v=kPRA0W1kECg

V této kapitole vytvoříme zoo a naplníme ji zvířaty z vašeho pole. Také vytvoříme přemisťovací pás, který vám umožní ukládat lokace na mapě a okamžitě se na ně teleportovat. Nakonec sami vytvoříte vlastní nezávislý projekt, který využívá pole obsahující barevnou vlnu, zvířata a příšery, zdobený pískovec či cokoli jiného!

Offline aktivita: Diskuse o polích v reálném životě

Otázky pro žáky:

- Zeptejte se svých žáků, jestli něco sbírají. Co je to? Komiksy, karty, mince, akční figurky, knížky atd.
- Jak velká je vaše sbírka?
- Jak je organizovaná?
- Jsou položky nějak roztříděné?
- Jakým způsobem byste našli určitý předmět z vaší sbírky?

V diskusi zkuste využít následující slova spojená s poli v kontextu sbírek vašich žáků:

- Délka pole: kompletní počet všech předmětů ve sbírce.
- Třídění: předměty ve sbírce seřazené podle nějakého přesného atributu (např.: datum, cena, jméno, barva atd.)

- Index: jedinečná adresa nebo umístění ve sbírce (např.: číslo strany v albu, polička v knihovně atd.)
- Typ: druh položky tříděné do sbírky (např.: DC komiksy, dolarové mince, karty Pokémon apod.)

Offline aktivita: Bublinové třídění (řazení)

V této aktivitě si prakticky vyzkoušíme jednu třídící metodu na vlastních žácích. Nepotřebujete k ní PC. Třídit budete vaše žáky, kteří budou stát v přední části místnosti. Pokud jste vy nebo vaši žáci zvědaví, jak vypadá takový algoritmus třídění v podobě kódu, naleznete na konci této aktivity kód bublinového třídění v MakeCode.

Potřebné materiály:

- 10 listů papíru očíslované 1–10 velkými písmeny

Vyberte deset žáků a postavte je před tabuli. Vyvolejte dalšího žáka, který bude tzv. třídič. Zamíchejte papírky s čísly a každému žákovi dejte jeden. Měli by papír držet tak, aby bylo jejich číslo viditelné pro zbytek třídy. Každý žák reprezentuje jednu hodnotu v poli.

Počáteční třídění

Řekněte třídiči, aby umístil žáky jednoho po druhém pomocí pokynů, kam jít, na správné místo. Jakmile jsou seřazení, položte žákům následující otázky:

- Jakým způsobem vás seřadil do správného pořadí?
- Vidíte nějaký vzorec?
- Co dělal?

Snažte se, aby byli žáci pokud možno co nejpřesnější při vysvětlování svých postupů. Někdy pomůže, když napíšete kroky na tabuli. Například:

- Nejdřív vybral prvního žáka v řadě a umístil ho na správné místo.
- Pak pokračoval k ostatním žákům a umístil je na správné místo.
- Nebo nejdřív šel k žákovi č. 1, pak č. 2, atd.

Pokud je to potřeba, požádejte o upřesnění: Co znamená, že je umístil na správné místo?

Například umístit někoho na správné místo znamená:

- Vzít jej na začátek řady a porovnat jeho číslo s prvním člověkem.
- Pokud je větší, posunout ho doprava.
- Pokračovat, dokud není číslo osoby větší, než číslo osoby vpravo.

Toto je dobrý okamžik pro zmínku skutečnosti, že lidé jsou mnohem chytřejší než počítače. Co nás napadne přirozeně (vzor porovnávání a třídění), to je pro počítač mnohem náročnější na pochopení – proto musíme být velice přesní a specifičtí při zadávání instrukcí počítači, aby věděl, jak položky v poli třídit.

Bublinkové třídění

Jeden ze základních třídících algoritmů se jmenuje Bublinkové třídění (Bubble sort), podle toho, že vyšší hodnoty „probublávají“ směrem ke konci řady.

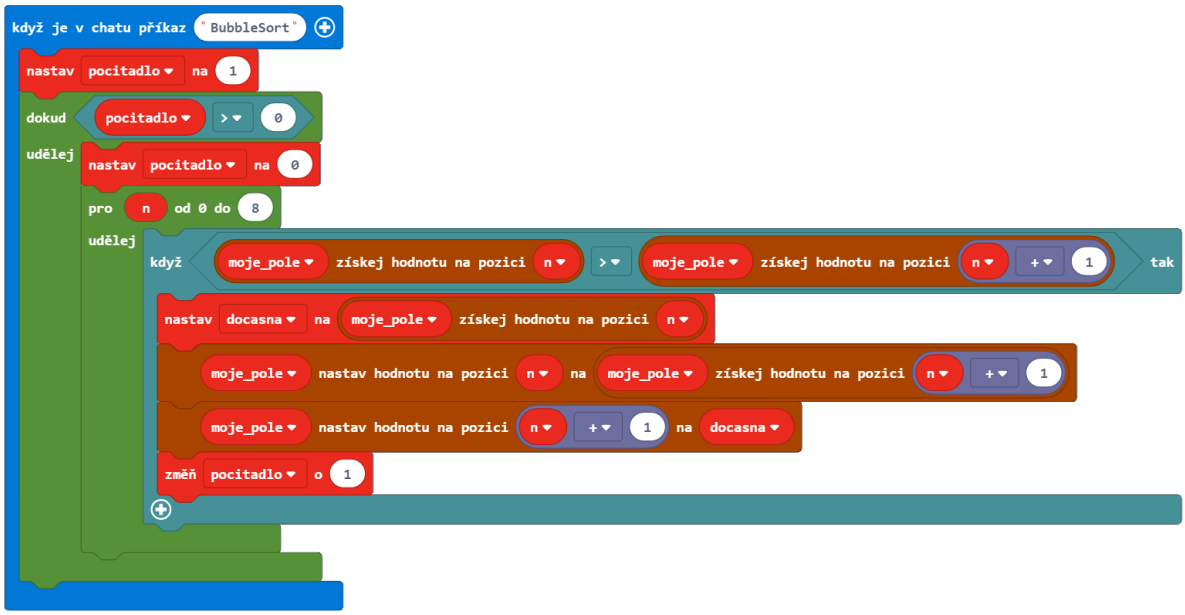
Jak si předvést bublinkové třídění:

- Porovnejte první dva žáky.
- Pokud je žák vpravo menší než žák vlevo, musí se prohodit.
- Pak porovnejte druhého a třetího žáka.
- Pokud je žák vpravo menší než žák vlevo, musí se prohodit.
- Když se dostanete na konec, začněte zase od začátku.
- Pokračujte stále stejným způsobem, dokud neprojdete celou řadou žáků a nikdo si nebude muset vyměnit své místo…

Pseudokód:

1. Vytvořte proměnnou jménem *počítadlo*.
2. Nastavte *počítadlo* na nulu.
3. Projděte celé pole.
4. Pokud je zvolená hodnota větší než hodnota napravo od ní:
    a. Vyměňte je.
    b. Zvětšete proměnnou *počítadlo* o 1.
5. Opakujte kroky 2 až 4 tak dlouho, dokud je hodnota proměnné *počítadlo* větší než nula.

Bublinkové třídění pomocí bloků v MakeCode:



V JavaScriptu:

```
let pocitadlo = 0
let docasna = 0
player.onChat("BubbleSort", function () {
    pocitadlo = 1
    while (pocitadlo > 0) {
```

```
        pocitadlo = 0
        for (let n = 0; n <= 8; n++) {
            let moje_pole: number[] = []
            if (moje_pole[n] > moje_pole[n + 1]) {
                docasna = moje_pole[n]
                moje_pole[n] = moje_pole[n + 1]
                moje_pole[n + 1] = docasna
                pocitadlo += 1
            }
        }
    }
})
```

Aktivita: Postavili jsme Zoo

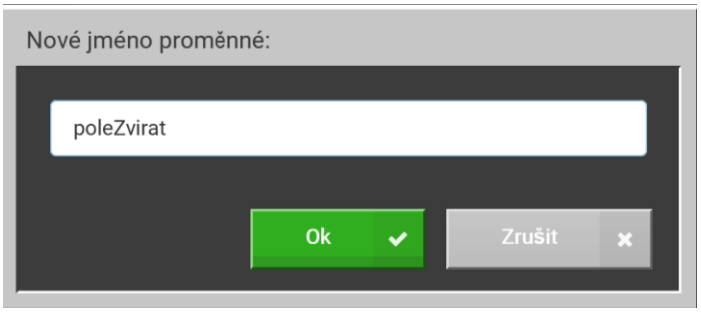
Zvířata můžete uchovávat v poli a umístit je, kamkoliv chcete. Využijeme tuto schopnost k postavení oploceného zvířecího výběhu a k vytvoření zoo. Na začátku projektu vytvoříme pole a naplníme ho zvířaty podle vlastní volby.

V tomto projektu použijeme dva příkazy:

- "vybeh": Tímto prvním příkazem povoláme Stavitele, abychom vytvořili oplocený výběh okolo své pozice, aby nám zvířata neutekla.
- "zoo": Tímto příkazem projdeme v MakeCode vytvořené pole a umístíme do výběhu dva jedince od každého druhu zvířat.

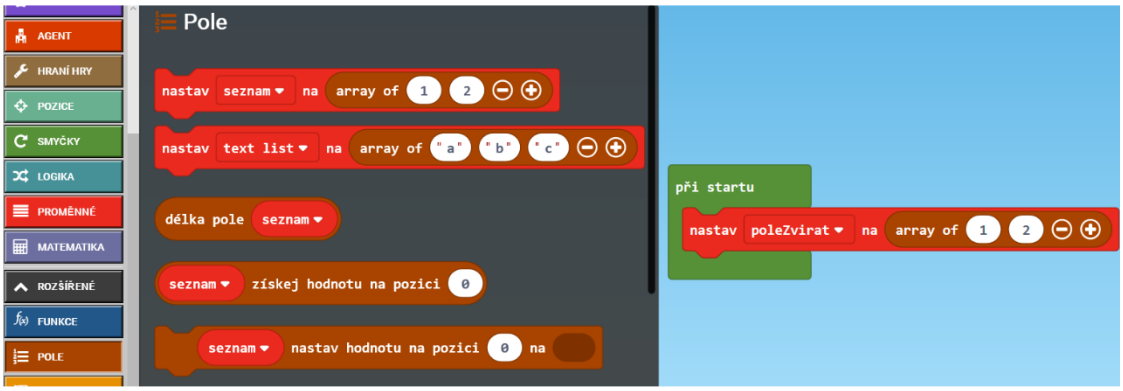
Kroky:

1. Vytvořte nový projekt v MakeCode s názvem „Zoo“.
2. V nabídce Smyčky naleznete blok Při startu, který automaticky spustí program, jakmile se přepnete do světa. Přetáhněte tento blok do pracovního prostoru.
3. V nabídce Proměnné klikněte na tlačítko „Vytvořit proměnnou“.
4. Pojmenujte proměnnou „poleZvirat“ a klikněte Ok.

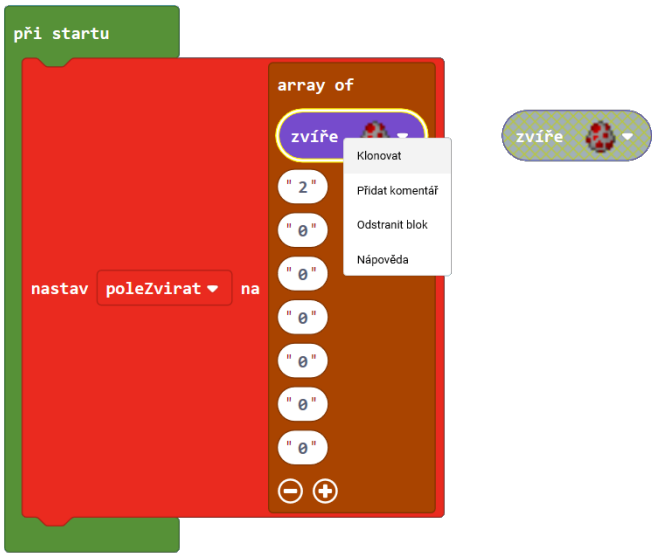


5. Klikněte na tlačítko Rozšířené v panelu nástrojů pro zobrazení záložky Pole.

6. Ze záložky Pole přetáhněte blok Nastav poleZvirat na dovnitř bloku Při startu.

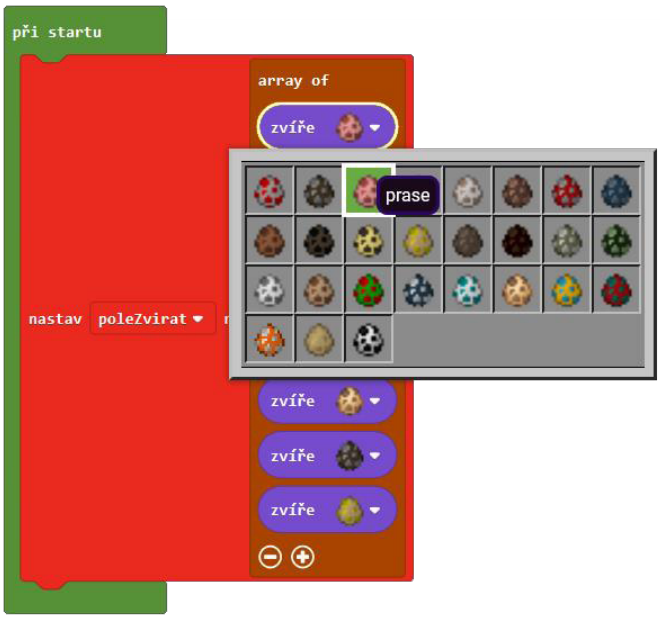


7. Klikněte na tlačítko plus (+) na bloku Array of a přidejte dalších 6 políček do svého pole. Celková délka vašeho pole bude 8.
8. Z nabídky Stvoření přetáhněte blok Zvíře do prvního slotu bloku Array of místo čísla 1.
9. Zaplňte zbytek pole bloky Zvíře. Poznámka – můžete kliknout pravým tlačítkem myši na blok Zvíře a vybrat Klonovat, abyste vytvořili kopie tohoto bloku a nemuseli je dokola přetahovat z panelu nástrojů.



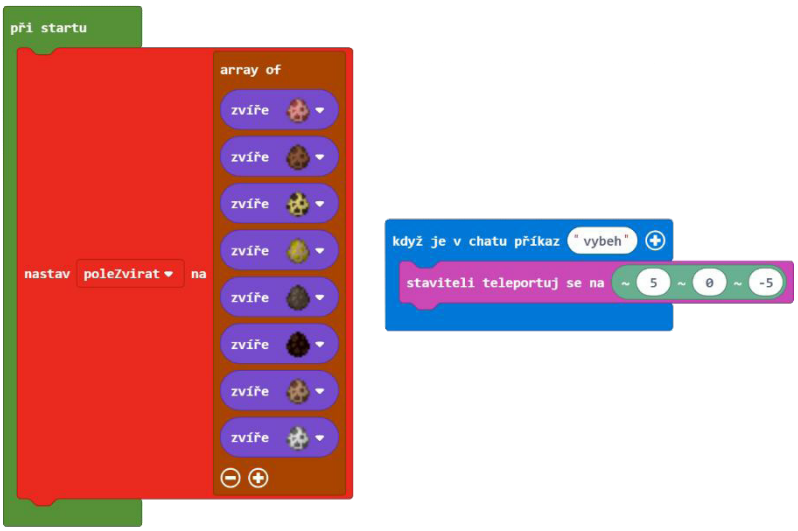
Vytvořte zoo s 8 různými druhy zvířat. Mějte na mysli, že některá zvířata mohou sežrat ostatní! Například oceloti a kuřata se nemají moc v lásce. Dobře si rozmyslete, jaký druh zoo chcete. Podle toho plánujte výběr zvířat.

10. Pomocí rozbalovací nabídky v blocích Zvíře vyberte různé druhy zvířat do vašeho pole.



Teď, když máme nastavené pole se zvířaty, můžeme zapracovat na oploceném výběhu pro naši zoo. K tomu budeme využívat bloky z nabídky Stavitel. Stavitel je jako neviditelný Agent ve hře, který může velice rychle pokládat bloky. Dáme stavitelovi pokyn, aby šel na bod v jihovýchodním rohu a položil zde počáteční značku (značka je neviditelná, Stavitel si tak uloží svoji výchozí pozici ve světě). Poté mu naprogramujeme sérii pokynů, podle kterých se bude pohybovat ve čtverci a stavět plot.

11. Z nabídky Hráč přetáhněte blok Když je v chatu příkaz.
12. Přejmenujte ho na "vybeh".
13. Klikněte na tlačítko Rozšířené v panelu nástrojů pro zobrazení karty Stavitel.
14. Z nabídky Stavitel přetáhněte blok Staviteli teleportuj se na do bloku Když je v chatu příkaz "vybeh".
    Vzpomeňte si, že souřadnice v Minecraftu jsou vždy určené souřadnicemi X, Y, Z, přičemž X je západo-východní směr a Z je severo-jižní. Chceme, aby Stavitel začal v severovýchodním rohu výběhu ve vztahu k hráči. Nastavíme souřadnice tak, aby určovaly oblast 5 bloků východně a 5 bloků severně od vaší pozice.
15. V bloku Staviteli teleportuj se na změňte hodnoty souřadnic na (~5, ~0, ~-5).



Ujistěte se, že Stavitel míří na správnou stranu, takže vytvoří výběh okolo vás. Položte značku začátku stavby.

16. Z nabídky Stavitel přetáhněte směrový blok Staviteli směr a položte ho pod blok Staviteli teleportuj se na.
17. Z nabídky Stavitel přetáhněte blok Staviteli umísti značku pod blok Staviteli směr.

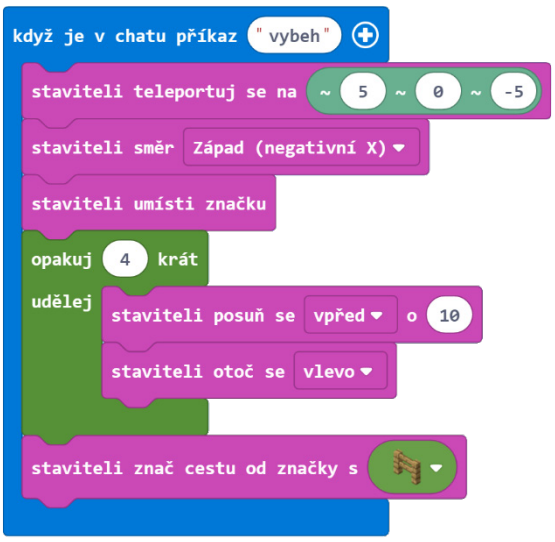
Nyní zadáme Staviteli instrukce, ať vytvoří čtvercový výběh.

18. Z nabídky Smyčky přetáhněte cyklus Opakuj a umístěte ho za blok Staviteli umísti značku.
19. Z nabídky Stavitel přetáhněte blok Staviteli posuň se dovnitř cyklu Opakuj.
20. Do bloku Staviteli posuň se zadejte číslo 10, abyste určili délku strany naší ohrady.
21. Z nabídky Stavitel přetáhněte blok Staviteli otoč se za blok Staviteli posuň se v cyklu Opakuj.

Poslední krok je, aby Stavitel po cestě pokládal plot.

22. Z nabídky Stavitel přetáhněte blok Staviteli znač cestu od značky s a vložte ho za cyklus Opakuj.
23. Z rozbalovací nabídky v bloku Staviteli znač cestu od značky s vyberte Dubový plot.

Váš kód by měl vypadat asi takto:



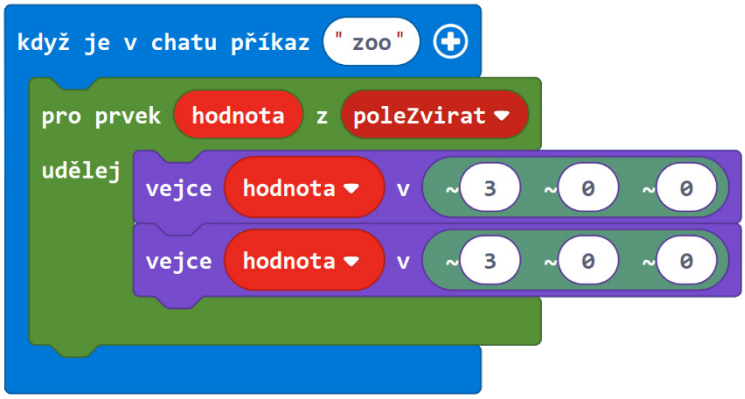
Otevřete v Minecraftu plochý svět a zadejte do chatu „vybeh“. Měli byste vidět, jak se kolem vás staví výběh! Jako vylepšení projektu můžete zařídit, aby Stavitel postavil i bránu.

Tady začíná zábava. Pole je naplněné zvířaty, výběh je hotový. Je na čase vypustit zvířata! Projdeme celé pole a pár bloků od vás, ale stále uvnitř výběhu, umístíme dva jedince od každého druhu zvířat.

24. Z nabídky Hráč přetáhněte blok Když je v chatu příkaz a přejmenujte jej na "zoo".
25. Z nabídky Smyčky přesuňte blok Pro prvek do bloku Když je v chatu příkaz "zoo".
26. V bloku Pro prvek vyberte z rozbalovací nabídky v druhém políčku možnost poleZvirat.

27. Z nabídky Stvoření přetáhněte blok Vejce dovnitř cyklu Pro prvek.
28. Z nabídky Proměnné přetáhněte proměnnou hodnota dovnitř bloku Vejce místo původního kuřete.
29. V bloku Vejce změňte pozici na (~3, ~0, ~0), takže se zvířata objeví několik bloků od vás.
30. Pro vytvoření párů klikněte pravým tlačítkem myši na blok Vejce a duplikujte jej.

Váš kód by měl vypadat zhruba takto:



Přepněte se zpět do světa Minecraftu, zadejte příkaz „zoo“ do chatu a sledujte, jak se zvířata objevují!

Celý program Zoo:

JavaScript:

```
player.onChat("vybeh", function () {
    builder.teleportTo(positions.create(5, 0, -5))
    builder.face(WEST)
    builder.mark()
    for (let index = 0; index < 4; index++) {
        builder.move(FORWARD, 10)
        builder.turn(LEFT_TURN)
    }
    builder.tracePath(OAK_FENCE)
})
player.onChat("zoo", function () {
    for (let hodnota of poleZvirat) {
        mobs.spawn(hodnota, positions.create(3, 0, 0))
        mobs.spawn(hodnota, positions.create(3, 0, 0))
    }
```

```
})
let poleZvirat: number[] = []
poleZvirat = [PIG, RABBIT, OCELOT, HORSE, DONKEY, MULE,
LLAMA, POLAR_BEAR]
```

Sdílený program: https://makecode.com/_2Jc55Xbjz1v8

Aktivita: Přemisťovací pás

Jedna z nejzábavnějších aktivit při hraní Minecraftu je stavba domů. Často je hráči budují rozeseté po celém světě. Někdy můžete během prozkoumávání narazit na vzdálených místech na úžasné skryté chrámy nebo vesnice. Může být náročné pamatovat si cestu zpátky na všechna tato místa. V této aktivitě vám ukážeme, jak vytvořit nástroj, který můžete využít k teleportování se mezi všemi těmito různými lokacemi.

V tomto programu budeme potřebovat tyto čtyři nové příkazy:

- **Smazat**: Vytvoří prázdné pole a odstraní to staré.
- **Uložit**: Toto uloží vaši stávající pozici do dalšího volného místa v poli.
- **Teleportovat**: Tento příkaz zadaný spolu s číslem teleportuje hráče na pozici uloženou v poli na tomto indexu.
- **Seznam**: Tento příkaz vypíše všechny pozice v poli spolu s jejich indexy.

Kroky:

1. V MakeCode vytvořte nový projekt s názvem „Teleportace“.
2. Z nabídky Smyčky vyberte blok Při startu a přemístěte jej do pracovního prostoru.
3. Otevřete nabídku Pole a přetáhněte blok Nastav seznam na array of.
4. Přejmenujte název proměnné seznam na poleLokaci.
5. Pomocí tlačítka mínus (-) odstraňte položky 1, 2 až vznikne prázdné pole.

Nyní vytvoříme příkaz pro smazání našeho pole. Když pole smažeme, budeme chtít automaticky vytvořit nové prázdné.

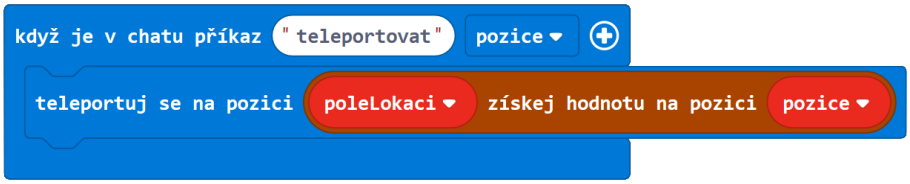
6. Z nabídky Hráč přetáhněte blok Když je v chatu příkaz a pojmenujte jej "smazat".
7. Klikněte pravým tlačítkem na blok Nastav ve vašem programu a vyberte Klonovat.
8. Přetáhněte blok Nastav dovnitř bloku Když je v chatu příkaz "smazat".
9. Z nabídky Hráč přetáhněte blok Řekni za blok Nastav.
10. V bloku Řekni zadejte text ve smyslu "Pole smazáno".

Abyste mohli uložit lokaci do pole, budeme chtít, aby příkaz Uložit přidal aktuální pozici na konec pole.

11. Z nabídky Hráč přetáhněte blok Když je v chatu příkaz a pojmenujte ho "ulozit".
12. Z nabídky Pole přetáhněte blok Přidej hodnotu na konec dovnitř Když je v chatu příkaz "ulozit".
13. Pomocí rozbalovací nabídky v bloku Přidej hodnotu na konec vyberte jako pole, do kterého chcete vkládat hodnoty, poleLokaci.
14. Z nabídky Hráč přetáhněte blok Pozice hráče ve světě dovnitř bloku Přidej hodnotu na konec.
15. Nakonec vyberte z nabídky Hráč blok Řekni a vložte jej za blok Přidej hodnotu na konec.
16. Do bloku Řekni zadejte text ve smyslu "Pozice byla přidána".

Hráč využije příkaz teleportovat ve spojení s číslem, např.: „teleportovat 1“ nebo „teleportovat 2“, aby se teleportoval na uloženou lokaci v poli.

17. Z nabídky Hráč přetáhněte do prostoru blok Když je v chatu příkaz a pojmenujte ho "teleportovat".
18. V bloku Když je v chatu příkaz klikněte na plus (+) pro přidání parametru num1. Ten bude reprezentovat index pozice v poli, kam se chcete přemístit.
19. Pomocí rozbalovací nabídky přejmenujte parametr num1 na pozice.
20. Z nabídky Hráč přetáhněte blok Teleportuj se na pozici dovnitř bloku Když je v chatu příkaz.
21. Z nabídky Pole přetáhněte blok Získej hodnotu na pozici do bloku Teleportuj se na pozici místo již existujícího bloku se souřadnicemi.
22. Pomocí rozbalovací nabídky v bloku Získej hodnotu na pozici vyberte proměnnou poleLokaci.
23. Z nabídky Proměnné přetáhněte blok proměnné pozice do druhého slotu v bloku Získej hodnotu na pozici.



Pokud nyní zavoláte příkaz teleportovat společně s číslem, teleportujete se na pozici uloženou v poli na dané pozici (indexu).

Možná vylepšení:

Existuje několik způsobů, kterými můžeme tento příkaz vylepšit. Nejdřív si uvědomte, že první položka v poli má index 0. Je trochu zvláštní zadat „teleportovat 0“ pro přesunutí na první pozici v poli. Upravme příkaz tak, že hráč může zadávat čísla od 1. Jediné, co musíme upravit, je odečíst 1 od proměnné pozice, abychom získali správný index. Tuto změnu uděláme v MakeCode editoru JavaScript. (Lze to však udělat i pomocí bloků.)

24. Klikněte na políčko JavaScript nahoře na obrazovce, aby se vám zobrazilo prostředí pro psaní kódu v JavaScriptu.

25. Najděte "teleportovat" funkci Player.onChat.
26. V následujícím řádku kódu odečtěte 1 od indexu tak, že řádek změníte na Player.teleport (poleLokaci [pozice - 1]).

```
player.onChat("teleportovat", function (pozice) {
    let poleLokaci: Position[] = []
    player.teleport(poleLokaci[pozice - 1])
})
```

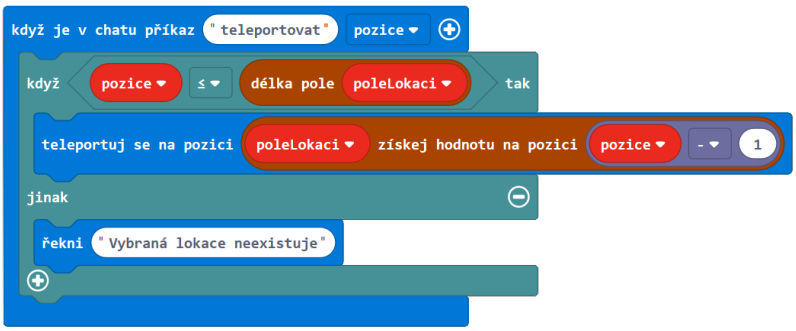
27. Pomocí tlačítka Bloky se vraťte zpět k editoru bloků

Tento program bychom mohli vylepšit ještě více. Pokud hráč zadá číslo větší než poslední index pole, mělo by se zobrazit nějaké varování. V opačném případě by mohl být hráč zmatený, proč se nikam neteleportoval. V programování je důležité dávat uživateli zpětnou vazbu o chybách a ošetřit všechny možné případy, které mohou nastat.

28. Z nabídky Logika vyberte blok Když Tak Jinak a vložte jej do bloku Když je v chatu příkaz "teleportovat".
29. Z nabídky Logika přetáhněte porovnávací blok Méně než (<) do prvního políčka bloku Když Tak Jinak místo pravda.
30. Z nabídky Proměnné přetáhněte blok proměnné pozice a vložte jej do porovnávacího bloku Méně než (<).
31. Z nabídky Pole přetáhněte blok Délka pole do druhého políčka bloku Méně než (<).
32. Z nabídky Proměnné přetáhněte blok proměnné poleLokaci dovnitř bloku Délka pole.
33. Pomocí rozbalovací nabídky v porovnávacím bloku Méně než (<) vyberte možnost Méně než nebo rovno (≤).

Pokud je index zadaný hráčem menší nebo roven délce pole, je vše v pořádku a hráč se přemístí na danou lokaci. V opačném případě (Jinak) by měl být hráč upozorněn chybovým hlášení.

34. Přemístěte existující blok Teleportuj se na pozici do první větve podmínky Když v bloku
    Když Tak Jinak.
35. Z nabídky Hráč na panelu nástrojů přetáhněte blok Řekni do podmínky Jinak bloku Když Tak Jinak.
36. Do bloku Řekni zadejte zprávu ve smyslu "Vybraná lokace neexistuje".



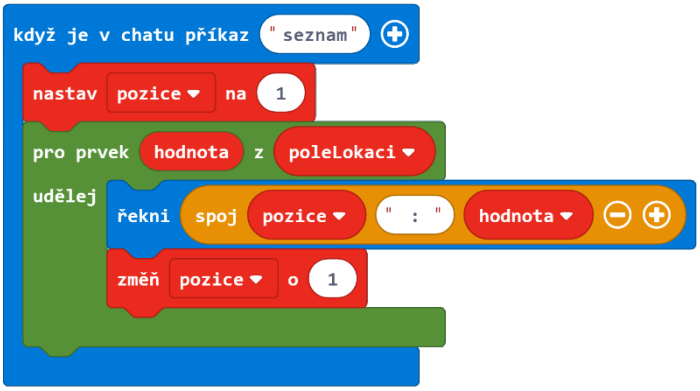
K dokonalosti chybí snad jen poslední příkaz! Vytvořme program, který vypíše všechny hodnoty v poli, abychom si je později mohli zaznamenat. Projdeme každým indexem v poli a vypíšeme jej spolu s odpovídající pozicí (hodnotou).

37. Z nabídky Hráč přetáhněte do pracovního prostoru blok Když je v chatu příkaz a pojmenujte ho "seznam".
38. Z nabídky Proměnné přetáhněte blok Nastav do bloku Když je v chatu příkaz.
39. Pomocí rozbalovací nabídky v bloku Nastav vyberte proměnnou pozice.
40. V bloku Nastav zadejte číslo 1, jako počáteční hodnotu proměnné pozice.
41. Z nabídky Smyčky přetáhněte cyklus Pro prvek za blok Nastav.
42. V bloku Pro prvek vyberte pomocí rozbalovací nabídky proměnnou poleLokaci.

43. Z nabídky Hráč přetáhněte blok Řekni do cyklu Pro prvek
44. V části Rozšířené najdete kartu Text. Z karty Text přetáhněte blok Spoj dovnitř bloku Řekni.
45. Z nabídky Proměnné přetáhněte proměnné pozice a hodnota do prvních dvou políček v bloku Spoj.
    Tímto se vypíše index pole spolu s jeho hodnotou, ale tato čísla budou vedle sebe bez mezery mezi nimi. Graficky tento výstup upravíme. Záměrně zvolíme způsob opět skrz JavaScript. (Opět existuje řešení i v blokovém prostředí pomocí tlačítka + příkazu Spoj).
46. Přepněte se do JavaScriptu a zobrazte svůj program v textové podobě.
47. Najděte funkci Player.onChat "seznam" a v ní řádek Player.say ("" + pozice + hodnota).
48. Přepište tento řádek kódu na Player.say (" " + pozice + " : " + hodnota). (včetně mezer okolo":").

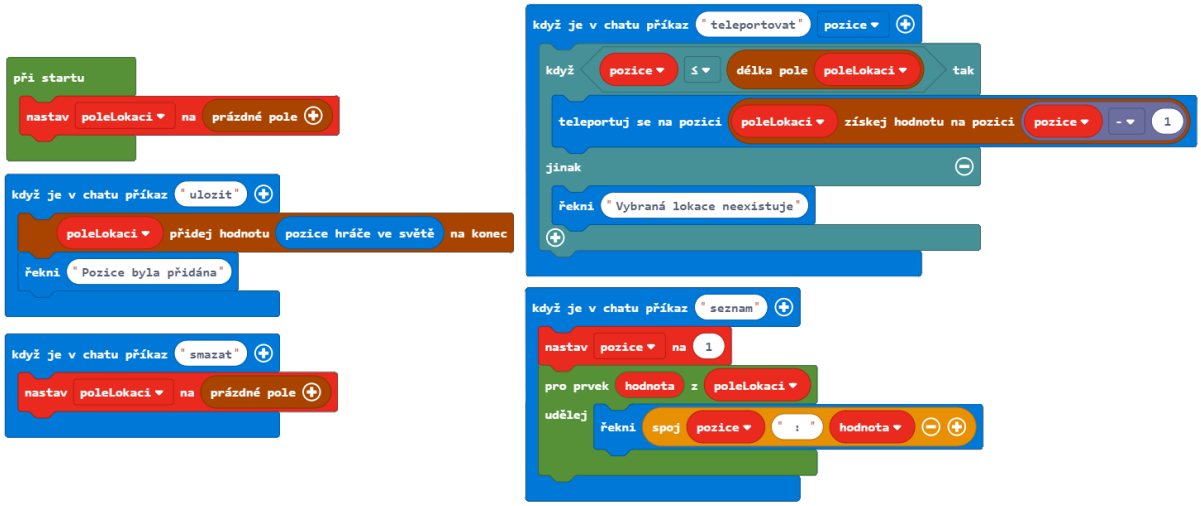
```
player.onChat("seznam", function () {
    pozice = 1
    for (let hodnota of poleLokaci) {
        player.say("" + pozice + " : " + hodnota)
    }
})
```

49. Přepněte se zpět do blokového prostředí.
50. Z nabídky Proměnné vložte blok Změň za blok Řekni.
51. V bloku Změň vyberte pomocí rozbalovací nabídky proměnnou pozice.



Teď máte svůj vlastní program pro zaznamenávání důležitých lokací ve vašem světě Minecraftu! Pamatujte, že ukončení programu nebo hry smaže celé pole se souřadnicemi. Mohlo by se hodit udělat si snímek obrazovky s vypsanými souřadnicemi nebo si napsat seznam všech svých souřadnic třeba do poznámkového bloku.

Kompletní program:



JavaScript:

```
player.onChat("seznam", function () {
    pozice = 1
    for (let hodnota of poleLokaci) {
        player.say("" + pozice + " : " + hodnota)
    }
})
player.onChat("ulozit", function () {
    poleLokaci.push(player.position())
    player.say("Pozice byla přidána")
})
player.onChat("smazat", function () {
    poleLokaci = []
})
player.onChat("teleportovat", function (pozice) {
```

```
    if (pozice <= poleLokaci.length) {
        player.teleport(poleLokaci[pozice - 1])
    } else {
        player.say("Vybraná lokace neexistuje")
    }
})
let pozice = 0
let poleLokaci: Position[] = []
poleLokaci = []
```

Sdílený program: https://makecode.com/_21x7XPK26Msx

Samostatný projekt: Umění s poli

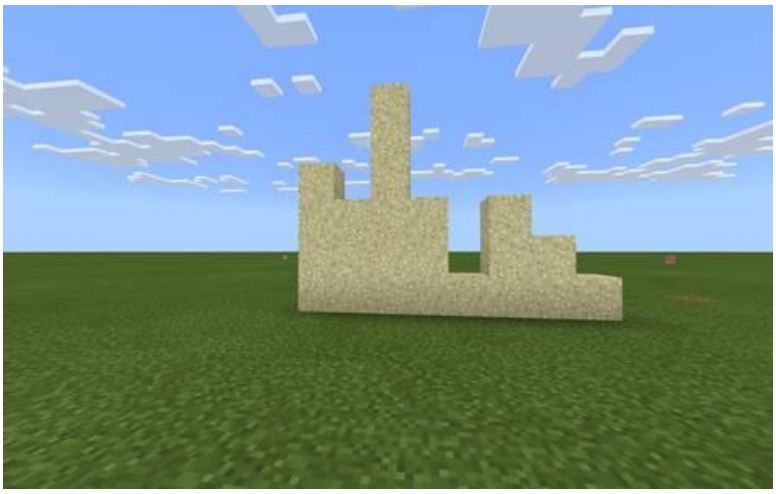
V samostatném projektu této kapitoly využijte pole pro vytvoření uměleckého díla. Pro inspiraci se podívejte na projekt s názvem Rainbow Beacon, který naleznete v tutoriálech na úvodní obrazovce MakeCode dole v sekci Super Powers. Tento projekt pracuje s polem s barevnou vlnou a pak z něj vystaví barevný sloup sahající až k nebi.

Jak byste mohli tento projekt modifikovat? Dokázali byste vytvořit duhovou cestu? Nebo postavit čtvercovou podlahu s duhovým kobercem? Pomocí polí a cyklů máte moc tvořit

velká umělecká díla. Vymyslete, jaký druh barevného, vícerozměrného uměleckého díla vytvoříte!

Některé nápady:

- Vytvořte návrhy a vzorce pokládáním různých druhů zdobeného pískovce uloženého v poli.
- Vytvořte sloupcový graf pouštěním různého množství písku z oblohy na základě čísla uloženého v poli.



- Vytvořte obrazec s využitím pole obsahující pouze bílou a černou vlnu.

Váš projekt by měl splňovat následující kritéria:

- Využívat pole.
- Přistupovat k položkám v poli buď sekvenčně (jeden po druhém) nebo náhodně.
- Přistupovat jen k indexům v rámci pole (ošetřit možnost, kdy zadáme index mimo pole).

Minecraft deník

Zapište si do deníku:

- Jaký druh umění jste se rozhodli tvořit? Co dělá váš program?
- Popište, jakým způsobem tvoří váš program dané dílo.
- Jak jste ošetřili, že je umožněn přístup jen k platným indexům?
- Vložte alespoň jeden snímek vašeho díla.
- Sdílejte svůj projekt na webu a zkopírujte URL odkaz.

POZNÁMKA: Pokud jste se rozhodli zlepšit jednu z aktivit této lekce, prosím, diskutujte se žáky o novém kódu, který jste napsali navíc k tomu, co již bylo poskytnuto v této lekci.

Hodnocení

	1	2	3	4
Deník	V deníku chybí odpovědi na 4	V deníku chybí odpovědi na 2	V deníku chybí odpovědi na 1	Deník obsahuje všechny

	nebo více otázek.	nebo 3 otázky	otázku.	odpovědi.
Projekt	Projektu schází všechny z požadovaných prvků.	Projektu schází 2 z požadovaných prvků.	Projektu schází 1 z požadovaných prvků	Projekt vytváří umělecké dílo účinně a efektivně.
Pole	Pole je špatně vytvořené nebo není vůbec zahrnuté.	Pole je vytvořené správně, některé prvky jsou nepřístupné, je možný přístup mimo hranice.	Pole je správně vytvořené, možné přistoupit ke všem prvkům nebo je možný přístup mimo hranice.	Pole je správně vytvořené, přístup ke všem prvkům, žádný přístup mimo hranice.

Umělá inteligence (UI)

V této kapitole se ponoříme do populárního tématu umělé inteligence (UI).

Od automobilů bez řidičů až po roboty, kteří porazili lidi v šachu, je oblast umělé inteligence jednou z nejvíce vzrušujících a slibných oblastí počítačové vědy. Umění tvořit programy, které napodobují a dokonce překonávají lidskou inteligenci, je nesmírně důležité. Společně s tím se ale objevuje celá řada etických otázek a obav. Na toto téma vzniklo mnoho sci-fi knih a futuristických filmů o strojích a robotech, které přebírají nadvládu nad člověkem a celým světem (Terminátor, Matrix, Já robot aj.).

Počítač Deep Blue od společnosti IBM byl první, který dokázal porazit člověka v šachu – v roce 1977 Deep Blue porazil světového šampióna Garryho Kasparova. Tato událost inspirovala filmaře k natočení dokumentárního filmu Game Over: Kasparov and the Machine.

Zde jsou některé nápady k zahájení třídní diskuze o umělé inteligenci:

1. Jaká kritéria musí počítač splňovat, abyste ho nazvali „inteligentní“?

   Neexistuje žádná definice, vědci se ale shodují na některých společných rysech:
   - Schopnost „inteligentních“ rozhodnutí
   - Schopnost učit se a zlepšovat své znalosti
   - Schopnost napodobovat lidi (jazyk/řeč, vidění/rozpoznávat obrazy)

   Počítačový vědec Alan Turing vymyslel tzv. Turingův Test, pomocí kterého se snažil určit, zda je stroj inteligentní.

2. Které věci by mohl počítač dělat, pokud bychom dokázali vytvořit dostatečně dobré programy?

   Některé příklady: diagnostikovat nemoci, řídit auta, pilotovat letadla, objednat za nás potraviny, vyprat, být naším osobním překladatelem na cestách, starat se nám o finance.

3. Je-li počítač inteligentní, znamená to, že má své vlastní vědomí, což znamená, je sám sebou?

UI nutně neznamená, že počítač může cítit nebo že má svou vlastní osobnost. Přesto mnoho výzkumníků studuje, zda bychom měli uvažovat nad právy robotů.

4. Jaké jsou obavy lidí ohledně UI? Jak skutečně si myslíte, že jsou tyto obavy opodstatněné?

   Některé příklady: roboti seberou lidem veškerou práci, počítače se stanou chytřejší než lidé a lidé se stanou jejich otroky, cyborgové se stanou vylepšenou rasou lidí.

Zajímavost: Výzkum umělé inteligence

Mnoho společností včetně Microsoftu investují do výzkumu a vývoje umělé inteligence. „Project Malmo" je výzkumný projekt, který se zabývá umělou inteligencí agentů v Minecraftu. Project Malmo je také mód v Minecraftu, který umožňuje počítačovým vědcům využívat prostředí Minecraftu k testování jejich umělé inteligence. Minecraft je ideální pro výzkum umělé inteligence, jelikož nabízí téměř neomezené možnosti, od jednoduchých úkolů jako je hledání pokladů, až po komplexní úkoly jako je skupinová stavba objektů.

Například si představte, že jste ve stavebnictví robot stavící budovy a snažíte se naučit, jak se dostat na kopec. Můžete to nejprve vyzkoušet v Minecraftu a předejít tak reálným případům, které vás mohou stát drahou opravu robota pokaždé, když například spadne do řeky. Robot začíná s tím, že neví nic o svém prostředí nebo dokonce o tom, čeho má dosáhnout. Musí porozumět svému okolí a rozhodnout, co je důležité – jít do kopce – a co není, jako třeba zda je den nebo noc. Musí zvládnout spoustu pokusů a omylů, jakými jsou pravidelné pády do řeky nebo lávy. A musí pochopit, kdy dosáhl cíle nebo alespoň jeho části.

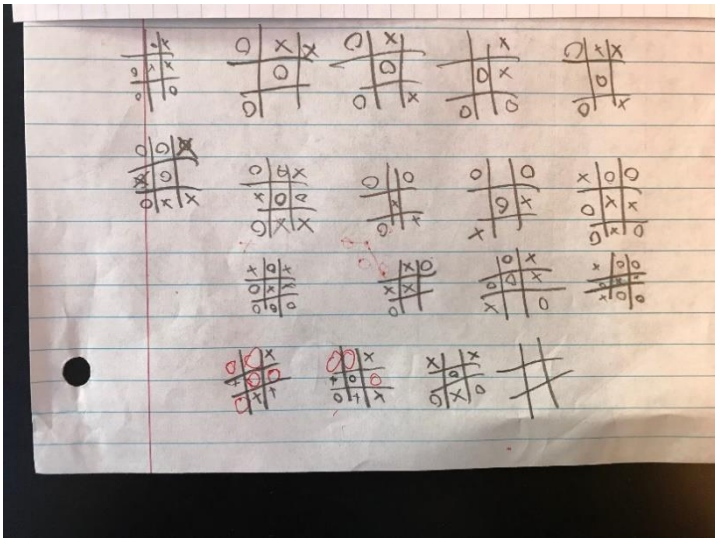
Offline aktivita: Papírová UI

Zahrajeme si se žáky hru, kterou velmi dobře znají. Piškvorky! Zjednodušíme si pro naše potřeby pravidla tak, že vítězem je ten, který propojí tři své symboly (křížky, kolečka) v hracím 3x3.

- Nechte žáky vytvořit dvojice a zahrát si několik kol piškvorek. Většinou si žáci hraní užívají, protože je to pravděpodobně již delší dobu, kdy naposledy piškvorky hráli.
- Zeptejte se studentů, jak moc přemýšleli nad jednotlivými rozhodnutími (tahy) - jak se rozhodovali, kam umístit křížek a jak jejich volbu ovlivňovaly tahy protihráče.

Většina studentů, kteří hru dobře znají, nad svými rozhodnutími nijak hluboce nepřemýšlí.

- Nechte je zahrát si několik dalších tahů, ale tentokrát se zaměřte na jejich rozhodování. Proč udělali tah právě takto, jak tah soupeře ovlivnil jejich tah?
- Společně si povídejte o rozhodnutích, kam umístit svoji značku.
- Požádejte je, aby vytvořili seznam pravidel/strategií, které by mohli být obecným návodem k vítězství ve hře piškvorky. Řekněte jim, že k popsání svých pravidel mohou využít struktury když-pak (Když-Jinak). To, co nyní vytvářejí, je tzv. pseudokód – mix běžného jazyka (češtiny) a kódu.
- Na úvod je dobré celé třídě napovědět s prvním pravidlem. Nechte je společně navrhnout pravidlo pro první tah. Většina z nich navrhne, že pokud je volné místo uprostřed hracího pole, umístěte svou značku právě tam.
- Poté ať se zamyslí nad dalším tahem a následně dalším atd. tak dlouho, dokud nevytvoří kompletní seznam pravidel/strategií, které by mohli pomoct komukoliv vyhrát.
- V této chvíli obvykle žáci zjistí, jak moc je nutné si promyslet každý svůj tah. Tato aktivita je pro většinu studentů skutečnou výzvou, a to i pro ty, kteří si na začátku mysleli, že sestavení pravidel bude jednoduché.
- Nakonec ať si žáci mezi sebou vymění svoje seznamy pravidel a zahrají si několik kol piškvorek proti sobě. Důležité při hře je, že každý smí použít pouze takové tahy, které odpovídají seznamu pravidel od jeho spolužáka. Nesmí používat pravidla a strategie, které vymysleli oni sami. Tuto aktivitu si většinou užijí a na základě této zkušenosti začnou více přemýšlet nad tím, jak upravit svá pravidla, abychom podle nich vždy vyhráli.
- Požádejte je, aby si představili, co je zapotřebí k naprogramování počítače pro hraní například šachů!
- Nepovinná výzva: Napište seznam pravidel/strategií pro přehrání hry typu „spoj 4 v řadě“. Vyzkoušejte to se spolužákem.



Hra piškvorky

Zde je několik příkladů návrhu pravidel pro umělou inteligenci ke hře piškvorky:

- Tah 0: umísti svou značku do středu hracího pole.

- Tah 1:
    - Jestliže tvůj protihráč umístil svou značku vedle tvé (mají společnou stranu), umísti svoji další značku diagonálně.
    - Jinak umísti svoji značku vedle tvé první.
- Tah 2:
    - Jestliže máš 2 značky vedle sebe v řádku, sloupci nebo diagonálně a třetí pole v tomto řádku, sloupci nebo diagonále je volné, umísti zde svoji značku, abys vyhrál.
    - Jinak umísti svoji značku do jiného rohu.

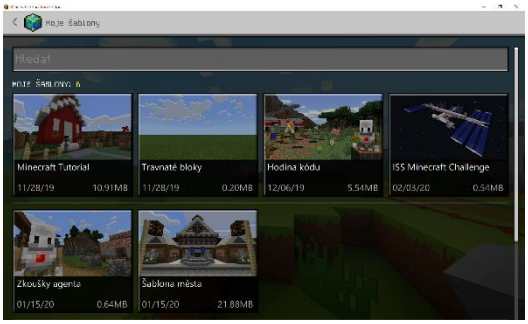
- Tah 3 a další:
    - Jestliže máš 2 značky vedle sebe v řádku, sloupci nebo diagonálně a třetí pole v tomto řádku, sloupci nebo diagonále je volné, umísti zde svoji značku, abys vyhrál.
    - Jinak jestliže má tvůj protihráč 2 značky vedle sebe v řádku, sloupci nebo diagonálně a třetí pole v tomto řádku, sloupci nebo diagonále je volné, zablokuj ho.
- Jinak umísti značku na libovolné prázdné pole

Aktivita: Vytvoření bludiště

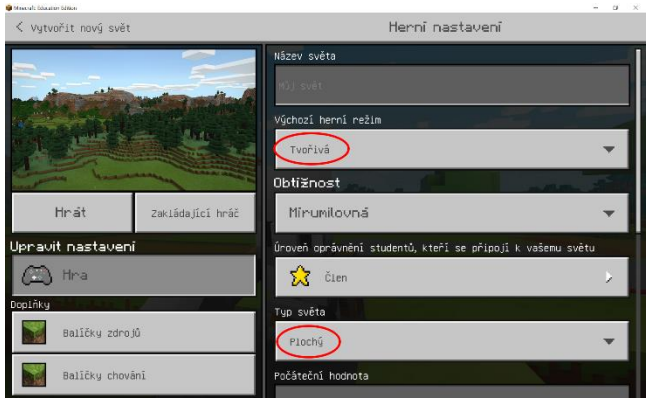
Naučit se číst a psát jde ruku v ruce. Stejně jako v běžném životě okoukáváme to, co dělají ostatní, je dobré sledovat programy jiných lidí a snažit se porozumět, jak fungují. Tímto způsobem se také můžeme zlepšovat v psaní vlastního kódu. V této aktivitě místo psaní vlastního kódu budeme zkoumat již existující kód pro vytvoření bludiště.

Kroky:

1. Spusťte Minecraft a vytvořte nový plochý svět a nastavte herní režim na tvořivý.

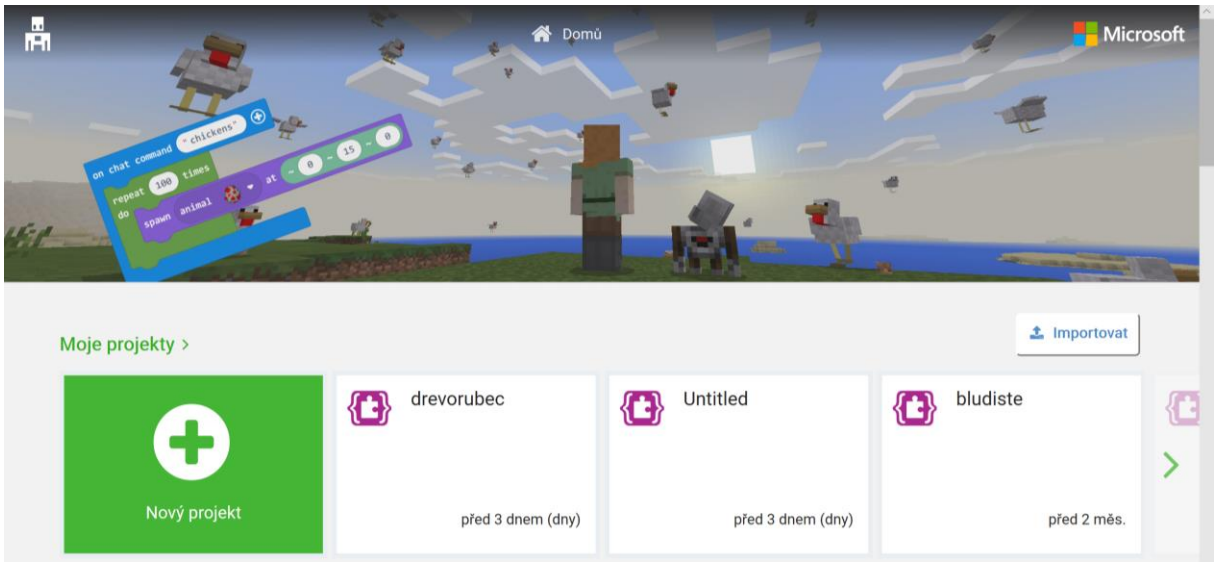


V Minecraft: Education Edition můžete použít šablonu Travnaté bloky

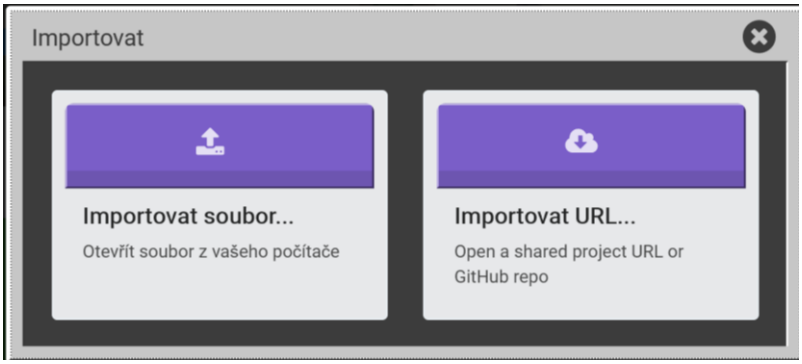


V Minecraftu ve Windows 10 vytvoř nový svět a nastav herní režim na tvořivý a typ světa na plochý

2. Otevři programovací prostředí MakeCode stisknutím klávesy C.
3. Stiskni tlačítko Importovat.



4. Zvol Importovat URL.



5. Zkopíruj a vlož tuto URL adresu: https://makecode.com/_h1zhdDCaDJMX
6. Do agentova inventáře na první pozici vlevo nahoře vlož kámen.

7. V Minecraftu stiskni klávesu T a do chatovacího okna napiš příkaz „maze“. Tím spustíš nahraný program.

Kód k vytvoření tohoto bludiště je trochu náročnější a raději napsaný v JavaScriptu než pomocí bloků. Ale zjednodušeně příkaz "maze" nejprve vytvoří okolo vaší pozice kamennou desku o velikosti 10 x 10 bloků. Funkce "dig" (kopej) naviguje Agenta kamennou deskou a ten pomocí příkazu pro ničení bloků vykope cestičky a tím vytvoří bludiště. Pokud se Agent prokope nějakou stěnou bludiště, postaví na jejím místě kamenný blok ze svého inventáře. Pomocí funkce "Shuffle" může Agent náhodně měnit směr a tím vytvořit různá bludiště.

```
player.onChat("maze", function () {
    blocks.fill(
    STONE,
    positions.create(-5, 0, -5),
    positions.create(5, 0, 5),
    FillOperation.Replace
    )
    agent.teleportToPlayer()
    blocks.place(AIR, positions.create(0, 0, 0))
    agent.setAssist(DESTROY_OBSTACLES, false)
    player.teleport(positions.create(0, 5, 0))
    player.say("let's dig that maze!")
    dig()
})
```

Otázky:

1. Popište, co dělá příkaz Blocks.fill.
2. Proč musíme umístit (Blocks.place) kostku vzduchu na pozici ~0 ~0 ~0?
3. Proč si myslíte, že jsme nastavili Agentovi příkaz znič překážky (DESTROY_OBSTACLES) na nepravda?
4. Jaký je rozdíl mezi oběma příkazy teleport?

Odpovědi:

1. Příkaz Blocks.fill vytvoří okolo hráče kamenný blok o velikosti 10 x 10. Agent bude z této kamenné desky vyřezávat bludiště.
2. Agent začíná uprostřed desky a prořezává se směrem ven, takže nejprve teleportujeme Agenta na pozici hráče, a poté v tomto místě vytvoříme pomocí bloku vzduchu díru.

3. Nastavili jsme „Znič překážku“ na nepravda, protože nechceme, aby Agent zničil bloky před ním, dokud je nebude mít možnost prozkoumat. To nám pomáhá určit, zda se nacházíme v již existujícím bludišti, nebo ne.
4. První příkaz pro teleport přemístí Agenta na pozici hráče, což je uprostřed bludiště. Druhý příkaz teleport přemístí hráče pět bloků výš nad bludiště, aby mohl sledovat celou tvorbu bludiště z ptačí perspektivy.

Tvorba bludiště je populární programovací úloha, jelikož existuje celá řada různých přístupů, jak dosáhnout cíle úlohy, stejně tak jako je možné vytvořit různé typy bludišť. Je také celkem zábavné sledovat, jak bludiště vzniká. Postup, který jsme použili, je jeden z nejpoužívanějších a je odvozen od algoritmu nazývaného „rekurzivní backtracking“.

Zde je pseudokód:

Zamíchej hodnoty v poli:

- Vytvoř pole obsahující tři směry
- Zajistěte jejich náhodné pořadí

Pro každou hodnotu v poli:

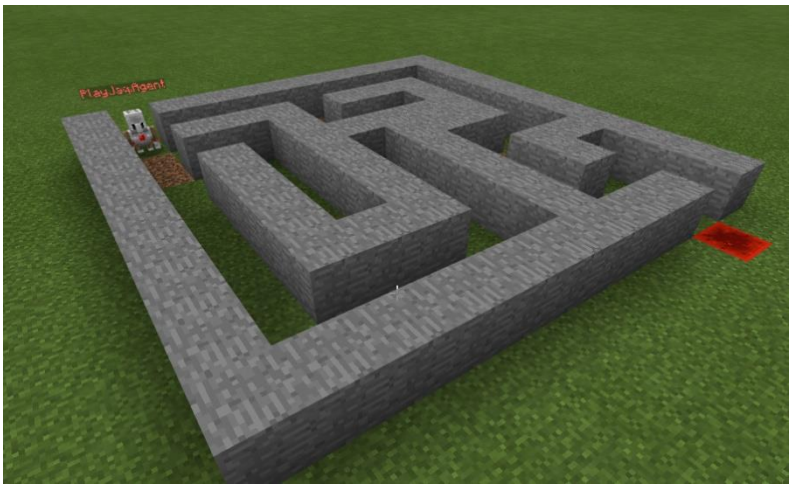
- Jestliže je hodnota vlevo, zahni vlevo
- Jestliže je hodnota vpravo, zahni vpravo
- Jestliže je před námi zeď (o šířce jednoho bloku), zachovej stěnu
- Jestliže před námi není zeď, kopej cestu dopředu (rekurzivně opakuj volání funkce "dig")
- Vrátíme se zpět o 2 bloky
- Otoč se zpět do původního směru

Hlavní myšlenka je taková, že Agent bude pokračovat v kopání nepravidelné cesty skrze kamenný blok tak dlouho, dokud nenarazí na prázdný prostor, přičemž se bude snažit nezničit žádnou ze zdí bludiště. Všimněte si, že na konci programu opět zavolá tu samou funkci a začne vyřezávat chodby znovu od začátku. Této speciální formě opakování se říká rekurze.

Aktivita: Cesta bludištěm

Nyní naučíme Agenta projít bludištěm. Spíš, než abychom mu dali sadu konkrétních příkazů typu „jdi 5 bloků rovně, zahni doleva“ atd., pomocí kterých by prošel pouze skrze jedno konkrétní bludiště, budeme se ho snažit naučit několik podmínek, díky kterým nalezne cestu skrze libovolné bludiště inteligentně.

Nejprve budeme potřebovat samotné bludiště. Buď můžete použít program z předchozího úkolu a nechat Agenta, aby pro vás nějaké bludiště vytvořil, nebo si můžete jedno postavit sami (což je docela zábava!). Agent musí poznat, kdy úspěšně došel až na konec bludiště. Proto označíme počáteční a koncový bod a umístíme ruditový blok pod východ z bludiště. Jakmile Agent rozpozná pod sebou rudit, zastaví se.



Algoritmus

V každém bludišti nebo labyrintu je obvykle možné najít cestu ven tak, že půjdete podél jedné zdi. Možná tímto způsobem nenajdete tu nejkratší cestu, ale dojdete k východu. V Minecraftím bludišti toto znamená, že kdykoliv to jde, zahněte vlevo. Pokud narazíte na slepou uličku, otočte se.

Jednoduchý algoritmus na průchod bludiště by mohl vypadat například takto:

- Opakuj tak dlouho, dokud nestojíš na ruditovém bloku:
  - Pokud nemáš po levé ruce žádný blok, otoč se doleva a udělej krok.
  - Jinak, pokud není před tebou žádný blok, udělej krok vpřed.
  - Jinak, pokud není blok po tvé pravé ruce, otoč se doprava a udělej krok.
  - Jinak se otoč a udělej krok vpřed.

Předtím než budeš pokračovat ve čtení v této knize, pokus se sám daný kód vytvořit.

Kroky:

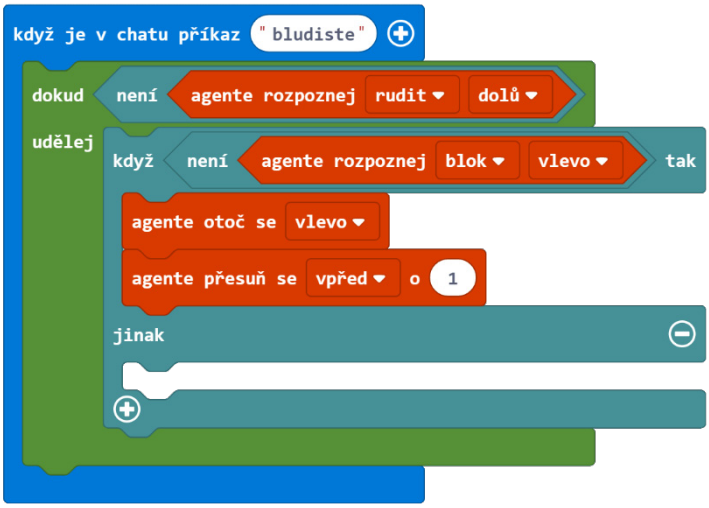
1. Vytvoř v MakeCode nový program s názvem „Hledání cesty“.
2. Přejmenuj již existující blok Když je v chatu příkaz na "bludiste".
   Celý program vložíme do smyčky s podmínkou, což znamená, že algoritmus se bude provádět tak dlouho, dokud Agent nevstoupí na ruditový blok.
3. Z nabídky Smyčky přetáhni blok Dokud pod blok Když je v chatu příkaz.
4. Z nabídky Logika vezmi blok Není a nahraď jím blok Pravda v podmínce smyčky Dokud.
5. Z nabídky Agent, přetáhni příkaz Agente rozpoznej dovnitř bloku Není.
6. V bloku Agente rozpoznej rozbal nabídku a vyber ruditový blok.
7. V bloku Agente rozpoznej rozbal nabídku a vyber směr dolů.

Tato část kódu znamená, že dokud pod sebou nebudeme mít rudit, bude se provádět kód uvnitř smyčky. Dále budeme zjišťovat, zda máme po své levé ruce volnou cestu, a pokud ano, otočíme se vlevo a popojdeme o krok vpřed.

8. Z nabídky Logika přetáhni blok Když Tak Jinak a umísti ho smyčky Dokud.
9. Z nabídky Logika přetáhni blok Není do Když Tak a nahraď blok Pravda.
10. Z nabídky Agent přetáhni blok Agente rozpoznej a vlož ho dovnitř bloku Není.
11. V bloku Agente rozpoznej rozbal nabídku a vyber možnost pohledu vlevo.

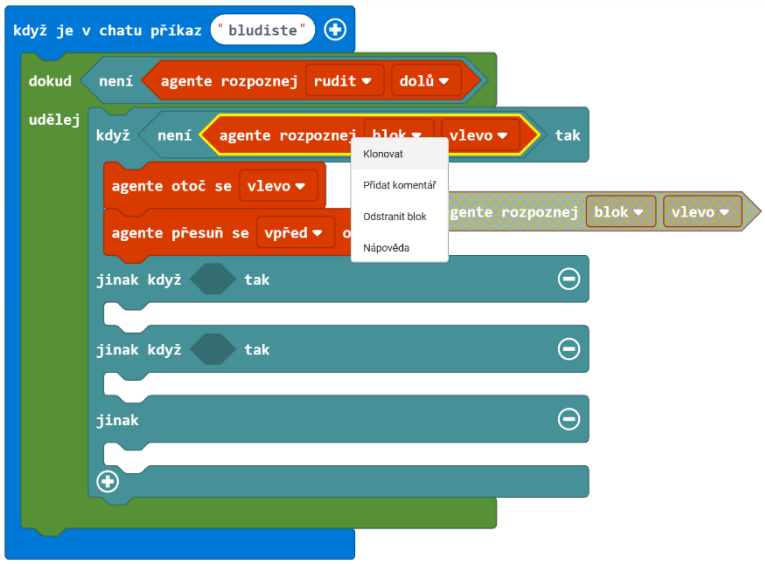
Jestliže Agent nezjistí blok po jeho levici (nedetekuje ho), otočí se vlevo a udělá krok vpřed.

12. Z nabídky Agent přetáhni bloky Agente otoč se a Agente přesuň se a vlož je do podmínky Když Tak

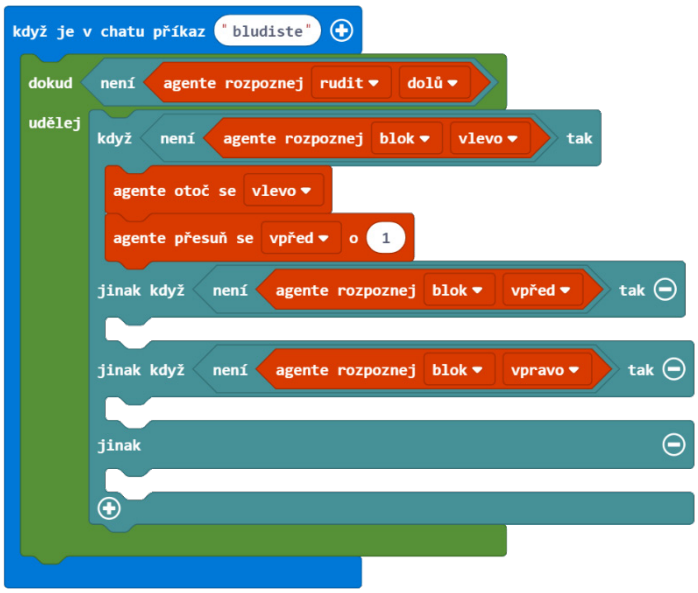


Nyní potřebujeme přidat 2 podmínky, abychom dokázali zjistit, zda se před Agentem nebo po jeho pravici nachází překážka.

13. Klikni dvakrát na znaménko (+) na konci bloku Když Tak Jinak, tím přidáš další dva případy Jinak Když.
14. Klikni pravým tlačítkem myši na blok Není v první části podmínky Když Tak a vyber Klonovat – tuto akci proveď dvakrát, abys získal další dvě podmínky Není Agente rozpoznej.



15. Vlož tyto Není Agente rozpoznej podmínky dovnitř dalších dvou otvorů bloků Jinak Když.
16. V prvním z těchto bloků Není Agente rozpoznej rozbal nabídku a vyber směr detekce dopředu.
17. V druhém bloku Není Agente rozpoznej rozbal nabídku a vyber směr detekce vpravo.

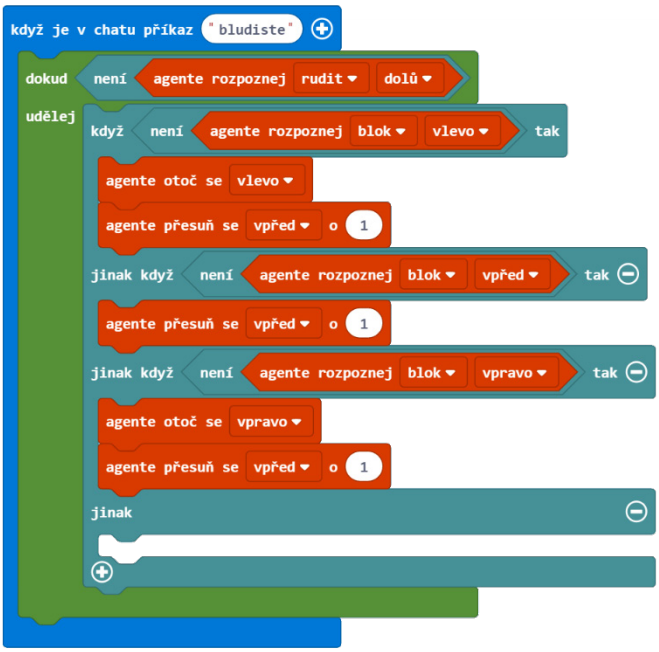


Jestliže Agent nemá před sebou blok, jednoduše může udělat krok vpřed.

18. Z nabídky Agent přetáhni blok Agente přesuň se do prvního bloku Jinak Když.

Jestliže Agent nemá blok po své pravé ruce, otočí se vpravo a udělá krok vpřed.

19. Z nabídky Agent přetáhni bloky Agente otoč se a Agente přesuň se do druhého bloku Jinak Když.
20. V bloku Agente otoč se vyberte z rozbalovací nabídky směr vpravo.

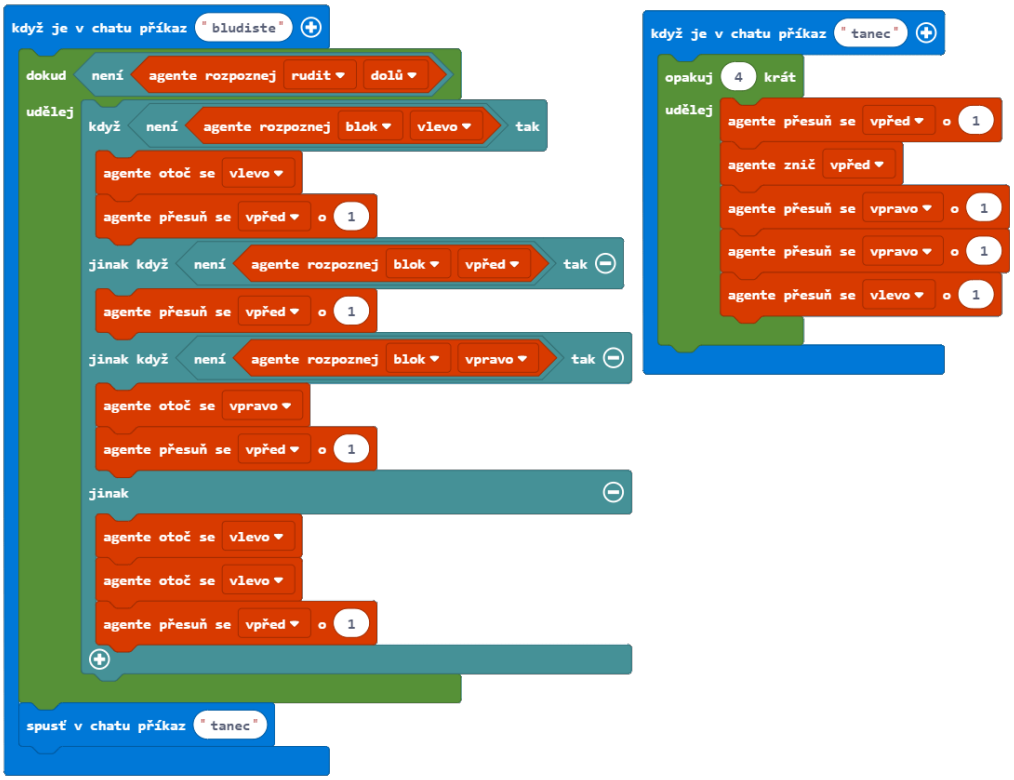


Pokud se Agent nakonec ocitne ve slepé uličce (žádný ze tří směrů není volný), chceme po něm, aby se otočil a vydal se směrem zpět.

Z nabídky Agent přetáhni následující 3 bloky do případu Jinak: 2x Agente otoč se blok a Agente přesuň se blok.

Jestliže Agent rozpozná pod sebou rudit, dosáhl cíle. Můžeme přidat blok Spusť v chatu příkaz "tancuj" a zatancovat vítězný taneček. Klidně můžete použít již hotový kód z 5. kapitoly (Cykly) z aktivity Tancuj, tancuj Agente.

Hotový program:



JavaScript:

```
player.onChat("tanec", function () {
    for (let index = 0; index < 4; index++) {
        agent.move(FORWARD, 1)
        agent.destroy(FORWARD)
        agent.move(RIGHT, 1)
        agent.move(RIGHT, 1)
        agent.move(LEFT, 1)
    }
})
player.onChat("bludiste", function () {
    while (!(agent.detect(AgentDetection.Redstone, DOWN)
)) {
        if (!(agent.detect(AgentDetection.Block, LEFT)))
 {
            agent.turn(LEFT_TURN)
```

```
            agent.move(FORWARD, 1)
        } else if (!(agent.detect(AgentDetection.Block,
FORWARD))) {
            agent.move(FORWARD, 1)
        } else if (!(agent.detect(AgentDetection.Block,
RIGHT))) {
            agent.turn(RIGHT_TURN)
            agent.move(FORWARD, 1)
        } else {
            agent.turn(LEFT_TURN)
            agent.turn(LEFT_TURN)
            agent.move(FORWARD, 1)
        }
    }
    player.runChatCommand("tanec")
})
```

Sdílený program: https://makecode.com/_K5TeAxH6e1zk

Skupinový projekt: Vytvoř UI

Poté, co máme za sebou samostatný projekt, je na čase pracovat se spolužáky a naučit Agenta nebo Stavitele inteligentně se přizpůsobit prostředí Minecraftu. Můžete použít podmínky, pomocí kterých Agent nebo Stavitel dokáže vyhodnocovat stav prostředí kolem něj a podle toho se sám rozhodovat.

V některých úlohách je výhodnější použít Stavitele místo Agenta. Stavitel je rychlejší a dokáže si pamatovat svoji současnou pozici. Ale jelikož je Stavitel neviditelný, je obtížnější odhalit případnou chybu, jelikož nevidíme, co přesně během své práce dělá.

Skupinová diskuze:

Nejprve si sedni se svým spolužákem a zkuste vymyslet pět až šest běžných úloh, které bychom mohli v Minecraftu automatizovat. Zde uvádíme několik příkladů:

- Hledání chrámů
- Hledání vesnic
- Hledání podzemních pevností (dungeon)
- Sklízení zralé pšenice
- Sklízení vzrostlé cukrové třtiny
- Hledání cesty z jeskyně

Se spolužákem vyberte dvě úlohy z vašeho seznamu, které vyžadují, aby Agent nebo Stavitel musel reagovat na své okolí. Pokuste se pomocí pseudokódu napsat kroky, které by vedly ke splnění těchto úloh.

Zde je opět příklad, jak najít v džungli chrám. Stavitel poletí nad korunami stromů a bude prozkoumávat džungli pod sebou, dokud nenarazí na mechový kámen položený na zemi, což je pravděpodobně část chrámu. Tento postup je velmi podobný tomu, který používají archeologové v Amazonii. Místo Stavitele používají v reálném světě radar.

Pseudokód:

- Vytvoř proměnnou pro kroky.
- Vytvoř proměnnou pro otočení.
- Přemísti Stavitele na tvou pozici.
- Vystoupej 20 bloků nahoru nad tvoji pozici.
- Pokud v oblasti mezi 15-20 bloky směrem dolů od Stavitele není mechový kámen:
    - Změň proměnnou kroky o 1
    - Jestliže počet kroků je větší než prohledávaná oblast:
        - Střídej zahnout vpravo nebo vlevo v závislosti na hodnotě proměnné otočka.
        - Nastav kroky na 0.
        - Změn hodnotu proměnné otočka (například, jestliže je pravda, změň ji na nepravda nebo obráceně).
    - Pohybuj se vpřed a prohledej další oblasti.
- Pokud jsi nalezl mechový kámen, ohlas svou současnou pozici, jelikož jsi pravděpodobně nalezl místo chrámu.

Podobným algoritmem lze hledat pevnosti nebo podzemní dungeony. Je vhodné nechat Stavitele informovat o jeho pozici v každém svém kroku (pomocí bloku Řekni). Máte tak přehled, kde se nachází a co dělá.

Ve dvojici rozhodněte, která z těchto úloh se vám zdá nejjednodušší a zkuste ji naprogramovat v MakeCode. Otestujte ji a upravte podle potřeby!

Minecraft deník

Zapište si do deníku:

- Jaký problém v Minecraftu ses rozhodnul řešit? Co tvůj program dělá?
- Popiš, jak se tvoje UI rozhoduje při řešení problému.
- Zamysli se a popiš rozdíly mezi samostatným projektem a způsobem práce na projektu ve dvojici.
- Popiš alespoň jeden bod, kde ses zasekl. Jak jsi to vyřešil?
- Vlož alespoň jeden snímek obrazovky, kde Agent nebo Stavitel provádí tvůj program.
- Sdílejte svůj projekt na webu a zkopírujte URL odkaz.

POZNÁMKA: Pokud jste se rozhodli některou z těchto aktivit vylepšit, promluvte si o novém kódu a porovnejte ho s tím, který je uvedený v této lekci.

Hodnocení

	1	2	3	4
Deník	V deníku chybí odpovědi na 4 nebo více otázek.	V deníku chybí odpovědi na 2 nebo 3 otázky	V deníku chybí odpovědi na 1 otázku.	Deník obsahuje všechny odpovědi.
Logické uvažování	Agent a/nebo Stavitel nikdy nedokončí přidělený úkol.	Agent a/nebo Stavitel splní svůj úkol jen v některých případech.	Agent a/nebo Stavitel většinou svůj úkol splní.	Agent a/nebo Stavitel vždy a za všech podmínek svůj úkol splní.
Projekt	Kód projektu je velmi těžkopádný a neefektivní.	Projekt řeší specifický problém, ale realizace úkolu je těžkopádná nebo neefektivní.	Projekt řeší daný problém většinou efektivně.	Projekt řeší daný problém účinně a efektivně.

Závěrečný projekt

V této závěrečné kapitole z MakeCode pro Minecraft necháme žáky vypracovat závěrečný projekt, který obsahuje dvě části:

1. Ukázat, co jsme se naučili
2. Demonstrovat něco nového

Kdybyste vyrazili vy a váš Agent na dobrodružství v Minecraftu a mohli byste Agenta naprogramovat na splnění pouze 4 úkolů, které by to byly? Žáci vytvoří sadu příkazů pro Agenta, kterých by mohl využít kdejaký dobrodruh, kdyby si ho vzal s sebou na cesty.

Kapitolu začneme opakováním konceptů, které jsme se naučili v minulých týdnech a navrhneme některé nápady, z kterých by se mohli žáci inspirovat při plnění finálového projektu v příštích hodinách. Poskytneme studentům nezbytné záchytné body, abychom je udrželi na daných úkolech a mohli sledovat výukové cíle.

Které nástroje byste si vzali do svého batůžku na dobrodružnou cestu Minecraftem?

Přehled kurzu

Zde je krátké shrnutí témat a pojmů, které tento kurz zahrnuje:

1. **Události**

   Kód je spuštěn událostí, která je výsledkem akce hráče. Zachytávač události sleduje různé akce, které se odehrávají ve hře, a spouští kód, který má spojen s těmito akcemi. Kód, který v Minecraftu napíšete, je spuštěn událostmi jako „když hráč jde“ nebo zavoláním vámi vytvořené funkce přes okno chatu.

2. **Souřadnice**

   Vaše pozice ve světě Minecraftu je dána třemi souřadnicemi představující pozici v trojrozměrném prostoru. Souřadnice mohou být určeny absolutně (nezávisle na vaší pozici) nebo relativně (vzhledem k současné pozici vaší postavy). Mnoho bloků v prostředí MakeCode používá souřadnice k určení polohy.

3. **Proměnné**

   Proměnnou používáme k uložení informace do paměti. Proměnné mohou být různých datových typů: čísla, text, boolean nebo pozice. Můžete změnit hodnotu proměnné nebo řídit jiné události na základě její hodnoty. Můžete také předat proměnnou jako parametr funkce volané v chatu.

4. **Opakování**

   Opakování nebo také cykly slouží k opakování instrukcí. Lze použít cykly s pevným počtem opakování nebo cykly řízené podmínkou. Můžete také vložit jednu smyčku do druhé k vytvoření složitějších opakování. Agent je tvůj přítel, který může v Minecraftu vykonávat instrukce za tebe.

5. **Podmínky**

   Podmínky používáme ve chvíli, kdy se potřebujeme v průběhu programu rozhodnout. Daný kód pak běží pouze tehdy, je-li splněna jedna nebo více podmínek. Použití bloků „Když ... Tak ... Jinak“ vám umožní zkontrolovat, zda je daná podmínka pravdivá a pokud ano, spustí se kód. V opačném případě se provede jiná sada instrukcí. Podmínky můžeme také negovat pomocí logického bloku „Není“.

6. **Funkce**

   Funkce jsou skupiny instrukcí. Jakmile definujete funkci, můžete se na tuto skupinu pokynů odkázat jednoduše podle jejího jména a tím spustit instrukce z těla funkce. Využívání názvů funkcí usnadňuje přehlednost a čitelnost složitých programů, které se tím stávají kratší. Skrytí jednotlivých pokynů ve prospěch obecnějšího názvu, který popisuje, co dělají, je příkladem abstrakce.

7. **Pole**

   Pole je způsob, jak uchovávat objekty. Můžete do něj ukládat věci a ty následně načíst dle potřeby v libovolném pořadí. Můžete ukládat čísla, text, pozice, dokonce i zvířata a různé bloky v Minecraftu. I když předem nevíte, kolik instancí objektů budete potřebovat uložit, jsou pole užitečná, protože vždy můžete přidávat nové objekty na konec pole.

8. **Umělá inteligence**

   Oblast umělé inteligence, zkráceně UI, rychle roste a nabízí mnoho příležitostí pro řešení problémů. UI může vašemu agentovi pomoci přizpůsobit se různým situacím ve světě Minecraftu a může vám umožnit psát kód pro mnoho různých možných scénářů. UI lze využít k vytváření složitých struktur nebo k navigaci stávajících. UI je velmi populární cesta, jakou se ubírá dnešní moderní kódování a programování ve světě.

Pro učitele

Závěrečný projekt je příležitost pro žáky, aby využili všechny znalosti, které se během pololetí naučili. Jejich úkolem je vytvořit sérii programů, které se dají využít ve světě Minecraftu. Časový rámec je flexibilní, ale měli byste počítat s tím, že žáci budou potřebovat několik hodin pro vymyšlení, testování a úpravy každé části kódu. Výsledkem by měly být tři nebo čtyři samostatné projekty.

Nejlepší projekty, zejména pak ty nejoriginálnější a nejkreativnější, nemusí být vždy projekty využívající nejsložitější kódy. Například některé originální nápady jsou jedinečné díky chytrým řešením, jako jsou např. zásobování záhonů vodou nebo tvorba promyšlené pasti pro neopatrné hráče. Samotný kód může být jednoduše několik bloků nebo kód spouštějící ruditové (elektrické) obvody.

Naopak jiné projekty nemusí zahrnovat stavění v Minecraftu. Mohou jednoduše zobrazovat výstup na obrazovku pomocí bloků Řekni. Žáci tím ale vyřešili konkrétní náročné programovací problémy jako například, jak zakódovat obraz do série polí.

Některé projekty budou obsahovat kód, který musí být spuštěný v určitém Minecraft světě. Jiné typy kódu zas mohou fungovat jen v plochém světě nebo mohou vyžadovat, abyste našli džungli nebo jiný druh terénu, abyste je mohli spustit. Proto většinou žádáme po žácích, aby

odevzdali krátké video svého kódu v akci, abychom jej mohli vidět tak, jak byl zamýšlen, aby fungoval.

Je důležité zastavit se a udělat krok zpět a podívat se plošně na celý projekt a rozpoznat, kde udělali žáci svou nejlepší práci, s čím měli největší potíže a na co jsou pyšní.

Vlastně spíše známkujeme proces toho, jak se během svého projektu žáci učili a zdokonalovali, než samotný výsledek. Svým žákům říkáme: „Nehodnotíš příběh podle jeho konce.“ Ve skutečnosti je často příběh tou nejzajímavější částí, a ne to, kde jste skončili. (Neřešíte tolik, že žili šťastně až do smrti. Spíše vás zajímají zkoušky a nesnáze, prostřednictvím nichž se hrdinové na konec dostali).

Řekněte žákům, aby se po dokončení projektu ohlédli a připomněli si ty okamžiky, kde na něco přišli, nebo si uvědomili chybu, či naopak dosáhli bodu zvratu v dějové lince svého řešení problému. Požádejte je, aby zvýraznili změnu v jejich kódu, která to reprezentuje, a aby stručně vysvětlili, čeho se to týkalo.

Jejich ohlédnutí je jako režisérův komentář „Making Of“ na DVD. Máte film a také komentář. Jejich vyprávění je jejich komentář. Nezabere jim to příliš dlouho, ale zvýší to uvědomění si, co se při práci na těchto problémech naučili.

Díky tomu můžeme známkovat to, co je nejvýznamnější: proces jejich učení a jejich chyb a oprav, ne dokončený kód, který je dokonale perfektní.

Další významný způsob, jak posoudit, jestli žáci dělají to, na čem záleží, je požádat je, aby navrhli projekty, které mají smysl nebo řeší specifický problém v Minecraftu. Návrh a programování jsou náročné práce, ale popisování problému a vytváření několika řešení je výborný způsob učení a úkol pochopitelný i pro ty děti, které nemusejí být od začátku kovanými programátory. Toto také pobízí žáky, aby pracovali spolu a navzájem si dávali užitečnou zpětnou vazbu.

Přemýšlení o projektech z hlediska návrhu může také motivovat děti, aby se o programování zajímali víc. Pokud nemají schopnosti si kód vytvořit sami, mohou se spojit s někým, kdo je má. Mohou pak být pouze v roli toho, kdo vymýšlí řešení. Schopnost naučit se pochopit další osoby, identifikovat potřeby druhých, navrhnout vlastního řešení, pracovat se souvislou zpětnou vazbou a neustálé opakování jsou dovednosti nepopiratelně velmi důležité i v mnoha jiných směrech a oborech.

Zadání závěrečného projektu: Batoh přeživšího

Závěrečný projekt je pro vás šance ukázat vše, co jste se toto pololetí naučili.

Představte si, že jdete do zbrusu nového světa v Minecraftu. Co uděláte jako první? Diskutujte se spolužákem.

Některé příklady:

- Sesbírat dřevo
- Sehnat vlnu
- Získat jídlo
- Postavit dům
- Natěžit uhlí

Pak si vyberte jednu nebo více z těchto činností, které se dají naprogramovat. Svého Agenta si můžete vzít s sebou, stejně tak Stavitele a ostatní mocné nástroje ve vašem MakeCode inventáři, ale jste omezení na jediný projekt. Jaké čtyři příkazy si vytvoříte?

Navíc musí váš projekt splňovat obě následující podmínky:

1. Ukázat něco, co jste se naučili.
   Měli byste prokázat své znalosti z jedné nebo více lekcí, které jste v průběhu pololetí probrali.
2. Ukázat něco nového.
   Měli byste vytvořit kód, který jste vymysleli sami a který nebyl dříve použit.

Časový rámec: Přibližně 4-5 vyučovacích hodin v závislosti na schopnostech žáků. Poslední hodinu byste měli věnovat testování. Projekty si žáci mohou testovat mezi sebou.

Každou hodinu vytvořte:

- 2–3 záznamy práce
- 1 záznam myšlenek

Některé možné nápady:

Potřeba	Řešení
Zábava	Vytvořte minihru
Mnoho papíru	Vytvořte a skliďte farmu na cukrovou třtinu
Lepší nástroje	Vytvořte hledač železa
Bezpečí	Vytvořte automatický přístřešek
Umění	Vytvořte něco krásného s použitím kódu

Konečné výstupy:

- Záznamy práce
- Záznamy myšlenek
- Finální kód projektu
- Příběh projektu
- Video vašeho projektu v akci
- Konečná ukázka projektu a jeho obhajoba

Hodnocení:

- 50 % proces (prvotní návrh, záznamy práce, záznamy myšlení, konečný příběh)
- 50 % produkt (sestavený kód projektu, videozáznam prováděného kódu)

Poznámka učitele: Tato forma hodnocení klade stejný důraz na záznam a návrh projektu jako na samotný dokončený produkt. Je to proto, že chceme, aby žáci pracovali na projektu průběžně, nikoliv jen poslední týden před odevzdáním. Takto budou nuceni průběžně dokumentovat svoji práci.

Nicméně se můžete rozhodnout přidělit každé části vyšší nebo nižší prioritu, aby odpovídala vaší třídě, stupni známkování a vašim prioritám při vyučování.

Při práci na projektu

Očekává se, že žáci pracují na svých nezávislých projektech pravidelně v průběhu tří týdnů a testují nápady, zkouší různé věci, zasekávají se a zase se dostávají dál. Protože každý pracuje na jiném projektu, nemůžeme všem zadat stejný domácí úkol. Kromě samotné práce na projektu jsou žáci zodpovědní za dokumentaci své práce pomocí záznamů a za zpracování svého procesu učení v záznamu přemýšlení. Zde je podrobnější ukázka:

1. **Pracovní záznam**

   Pracovní záznam je krátký odrážkový seznam toho, na čem jste pracovali a jak dlouho to zabralo. Držte se faktů. Zápis pracovního záznamu by neměl zabrat více než 30 sekund. Žáci by měli vytvořit jeden vstup do pracovního záznamu pro každou vyučovací hodinu. Sdílený sešit Microsoft OneNote je skvělý způsob, jak udržovat pracovní záznam, který mohou všichni žáci pravidelně aktualizovat. Jinak můžete používat libovolný sdílený dokument.

   Vzorový pracovní záznam:

   Projekt Batoh: Systém prozkoumávání jeskyní
   31. 3.: Práce na navigaci Stavitele skrz nepravidelné chodby (45 min.)
   1. 4.: Přidávání bloků Řekni do každého bodu, aby se pořád neztrácel (30 min.)
   3. 4.: Dokončení prozkoumávání jednoduchých jeskyní, momentálně práce na složitějších soustavách (45 min.)
   4. 4.: Kódování víceúrovňových jeskyní, systém detekce líhní oblud (30 min.)

   Poznámka pro učitele: je dobré obecně neuznávat pozdní pracovní záznamy. Pokud žák chyběl, měl by pořád sestavit pracovní záznam a zapsat, že tuto hodinu nepracoval. Pracovní záznamy nemají takovou váhu, takže vynechání jednoho nebo dvou není problém. Pokud mu ale chybí více záznamů, je na čase zkontrolovat, jak daleko se žák se svým projektem dostal.

2. **Záznam myšlení**

   Záznam myšlení je jako vstup do deníku (podobný tomu, který jste tvořili na konci každé lekce), v kterém je zapsáno, co jste se během posledního týdne naučili. Projděte si svou práci za minulý týden a podívejte se na to, co jste udělali, kde jste se zasekli a jak jste to vyřešili. Na konci celého projektu sepište krátkou slohovou práci, v které odpovíte na následující body:

   - Popište něco, co vás překvapilo při práci na projektu.
   - Popište okamžik, kdy jste se zasekli. Jak jste se odtamtud dostali?
   - Pomáhal vám někdo? Kdo a jak?
   - Vyberte přídavné jméno, které popisuje váš pocit z projektu. Vysvětlete, proč jste vybrali toto slovo.
   - Kdybyste měli více času na práci na tomto projektu, co byste přidali (vylepšili)?

   Závěrečnou práci mohou žáci sepisovat klidně průběžně.

   Ukázka ze záznamu myšlení:

   Týden 6. dubna

Konečně jsem přišel na to, proč mi můj kód nefungoval. Měl jsem nesprávný cyklus vložený do jiného, takže se spouštěl mnohem víckrát, než měl. Agent kvůli tomu scházel z cesty. Jakmile jsem vytvořil další proměnnou, abych měl přehled, kolikrát se Agent vrátí do první řady, a během každého cyklu jsem tuto proměnnou kontroloval, všechno fungovalo tak, jak mělo. Tohle mě napadlo jedině díky tomu, že jsem postupně vypínal jednu funkci po druhé, abych vystopoval problém. To bylo užitečné. Příště si musím pamatovat, abych to vyzkoušel.

Poznámka učitele: Záznam myšlení *není* prodloužený pracovní záznam! Žáci někdy prostě jen napíší podrobnější seznam všech úkolů, které v týdnu splnili, ale to není smysl Záznamu myšlení. Pracovní záznamy mají ukázat, CO jste dělali. Záznam myšlení má ukázat, JAK jste se to naučili dělat. Na rozdíl od pracovních záznamů, můžete přijímat pozdní Záznamy myšlení, pokud přijdou před datem pro další Záznam myšlení. Je to důležitá forma dokumentace učebního procesu.

Co odevzdat

Když odevzdáváte závěrečný projekt, měli byste odevzdat svůj kód a konečný zápis. Pro odevzdání kódu můžete svůj projekt sdílet přes tlačítko Sdílet v horní nabídce:

Potvrdíte, že souhlasíte se sdílením svého projektu tím, že kliknete na možnost „Publikovat projekt“ a pak vám MakeCode zobrazí jedinečný URL odkaz, který si můžete zkopírovat a odevzdat jako kód vašeho projektu.

Také musíte vytvořit závěrečný zápis, který odevzdáte spolu s kódem. Posledních několik hodin jste pracovali na vymýšlení, návrhu a testování originálního batohu s Minecraft kódem. Čekám upřímné a přesné hodnocení vaší práce za toto období.

Prosím, vraťte se zpět a pročtěte si všechny své pracovní záznamy a záznamy myšlení. Pak sestavte zprávu, v které popíšete celý průběh vašeho projektu. Jak tuto zprávu zpracujete, je na vás, ale mohli byste zahrnout odpovědi na následující otázky:

- Jak jste začali s návrhem projektu?
- Co jste se chtěli naučit?
- Jakým výzvám jste čelili? Jak jste je překonali?
- Co byl výsledek?
- Co jste se nakonec naučili?
- Kdo ve třídě vám v průběhu projektu pomáhal? Jak?
- Na co jste nejpyšnější?
- Co byste příště udělali jinak?

V závěrečné zprávě se musíte odkazovat na pracovní záznamy a záznamy myšlení (např.: Záznam myšlení 17. 4., Pracovní záznam č. 3, atd.) K tomuto můžete využít poznámky pod čarou nebo odkazy na citovaný materiál napsat do závorek.

Zprávu budu pozorně číst a známkovat spolu s vaším konečným kódem, záznamy práce a myšlení. Výsledek bude konečná známka projektu.

Ukázka závěrečné zprávy:

Teď je mi jasné, že v druhém týdnu jsem byla trochu ztracená a zmatená ze směru, kam se můj projekt dostával. Už mi došlo, že při diskusi s panem učitelem a s mými spolužáky (Zápisky 3. 4.) jsem se dostatečně neptala a že jsem úplně nepochopila, co se vysvětlovalo.

Když se ohlédnu, jeden z mých cílů byl, abych se pravidelněji radila s mými spolužáky z lavice. Snad bych byla mnohem méně ztracená a alespoň tentokrát, když bych se zasekla, bychom to vyřešili společně. Psala jsem o tom v mém Záznamu myšlení (Záznam myšlení #2). Jsem překvapená, že po setkání se spolužáky mi začalo spoustu věcí docházet. Toto mi umožnilo, abych se dostala přes něco, s čím jsem měla opravdu potíže, konkrétně přidávání a odstraňování věcí z pole, a byla jsem schopná to dokončit v další hodině. (Pracovní zápis #4)

Jakmile mi začalo být jasnější, co dělat, byla jsem schopná sehnat mnohem účinnější pomoc od spolužáků. Konkrétně Honza mi hodně pomohl vyřešit, jak využívat různé možnosti maskování v bloku Klonovat. Taky mi ukázal nějaké šikovné vzorové projekty a příklady v MakeCode. Myslím, že kdybych tohle mohla dělat ještě jednou od začátku, naplánovala bych si více času s panem učitelem a/nebo našla lepší způsob sdílení různých částí kódu s mými spolužáky, protože jsme se všichni snažili vyřešit zhruba ty samé problémy, jen v jiných projektech. Ani jsem až do konce projektu nevěděla, že se dá přeskočit do JavaScriptu a změnit kód, přičemž se z toho pak vytvoří správné bloky, když se vrátím! (Zápisky Beta Testing) To by mi bylo zachránilo hodně času.

K vyprávění, prosím, poskytněte krátké (méně než 2 minuty) video vašeho kódu v akci.

Beta testing, konečná ukázka a hodnocení

Beta testing je důležitá část testování konečných projektů, abyste odhalili chyby nebo problémy. Jedna možnost, jak testovat projekty, je požádat žáky, aby si je otestovali navzájem mezi sebou. Testování by nemělo probíhat v den odevzdání, ale projekty by měly být víceméně hotové (všechny funkce musí být zpracované do kódu a stavba jakýchkoli prvků projektu v Minecraftu musí být úplně nebo skoro hotová).

Žáci se mohou střídat v prezentaci svých projektů celé třídě, nebo mohou pracovat ve dvojicích, zkoušet projekty svých spolužáků a nabízet zpětnou vazbu. Žáci, kteří obdrží kritiku, by si měli dělat zápisky ze zpětné vazby a odevzdat je v záznamu o svém závěrečném projektu.

Konečná ukázka

Oslavte tvrdou dřinu svých žáků a uspořádejte ve škole prezentaci pro rodiče, vedení školy a další členy společnosti, abyste patřičně ocenili všechnu namáhavou práci, která byla vložena do každého ze závěrečných projektů.

Zjistili jsme, že formát „science fair“ funguje hezky. Žáci sedí u stolů, kde předvádějí své projekty, ukazují videa a odpovídají na otázky. Také byste mohli zvážit vytvořit si YouTube kanál, kde mohou vaši žáci zveřejnit návody na tvorbu různých příkladů kódu pro Minecraft, se kterými přišli, a přidat odkaz na sdílený kód. Tak či tak, trocha veřejného uznání pro vaše žáky za jejich tvrdou práci je rozhodně zasloužená!

Hodnocení

	1	2	3	4
Kód (Ukaž, co ses naučil)	Kód nedemonstruje využití předchozích konceptů a/nebo jména proměnných jsou nejasná. Je množství míst, kde by se kód dal zlepšit.	Kód demonstruje využití předchozích konceptů s minimální účinností. Jména proměnných mohou být nejasná. Kód by mohl být účinnější.	Kód víceméně účinně demonstruje využití předchozích konceptů. Jména proměnných jsou obecně jedinečná a přesně popisují, jakou hodnotu proměnné vyjadřují. Kód je poměrně účinný.	Kód velice účinně demonstruje využití předchozích konceptů. Jména proměnných jsou obecně jedinečná a přesně popisují, jakou hodnotu proměnné vyjadřují. Kód je vysoce účinný.
Kód (Ukaž něco nového)	Kód nedemonstruje žádné nové koncepty a/nebo jména proměnných jsou nejasná či neúčinná. Je množství míst, kde by se kód dal zlepšit.	Kód demonstruje využití nových konceptů s minimální účinností. Jména proměnných mohou být nejasná. Kód by mohl být účinnější.	Kód víceméně účinně demonstruje využití nových konceptů. Jména proměnných jsou obecně jedinečná a přesně popisují, jakou hodnotu proměnné vyjadřují. Kód je poměrně účinný.	Kód velice účinně demonstruje využití nových konceptů. Jména proměnných jsou obecně jedinečná a přesně popisují, jakou hodnotu proměnné vyjadřují. Kód je vysoce účinný.
Pracovní záznamy	Více než dva pozdě odevzdané či chybící a/nebo nepřesné či nedostatečné záznamy.	Dva pozdě odevzdané či chybící záznamy a/nebo záznamy nepřesné či nedostatečně podrobné.	Jeden pozdě odevzdaný či chybící a/nebo záznamy nepřesně či nedostatečně podrobné.	Všechny záznamy odevzdané včas a přesné.

Sebehodnocení	Reflektivní části schází 3 žádané prvky.	Reflektivní části schází 2 žádané prvky.	Reflektivní části chybí 1 žádaný prvek.	Reflektivní část popisuje • Proces vývoje • Něco nového • Něco, na co jsem hrdý • Budoucí doplňky

Rozšiřující příklady

Tyto příklady již nejsou součástí původní knihy, ale byly přidány autorem překladu. Úlohy jsou komplexnějšího charakteru a mohou sloužit buď jako doplňující úlohy k některým kapitolám, nebo jako inspirace pro závěrečné komplexní programy. Další příklady budeme postupně přidávat na stránky http://minecraftedu.cz

Programování Agenta: Bludiště 2

Zadání:

V této úloze naprogramujete Agenta, aby dokázal projít vytvořené bludiště. Algoritmus by měl být obecný, proto by Agentovi nemělo vadit, pokud se bludiště změní.

Řešení:

Možných řešení je určitě více. Žáci pravděpodobně jako první naprogramují Agenta pomocí přesné sekvence příkazů. Zde se nabízí rozšíření úlohy. Každý student postaví vlastní bludiště, vyexportuje svět a pošle ho spolužákovi. Pokud Agent projde i toto bludiště, je algoritmus s velkou pravděpodobností obecný. Svět nemusíte nutně exportovat, žáci mohou pracovat rovnou v jednom světě.

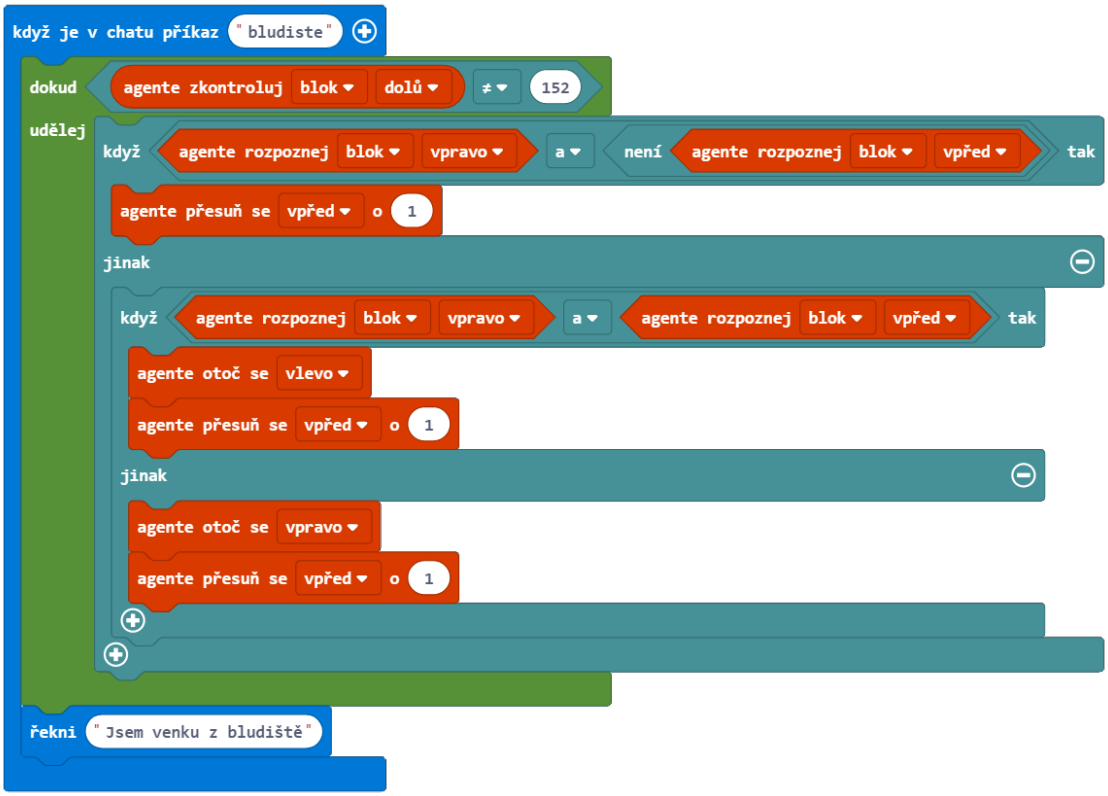
Obecné řešení:

Jedním z možných algoritmů, jak projít bludiště, je držet se jedné zdi. V mém algoritmu se Agent vždy drží zdi pravou rukou.

Stáhněte si svět Bludiště. Teleportujte Agenta na modré pole tak, aby měl zeď po své pravé ruce. Natočte ho směrem do bludiště. Algoritmus bude probíhat tak dlouho, dokud Agent nedojde na konec bludiště. Abychom zjistili, že je venku, zjišťuje Agent v každém svém kroku, jaké pole má pod sebou. Poslední bloky u východu jsou červené a záměrně jedinečné v celém bludišti. Agent dokáže zjistit tzv. ID bloku. Je nutné si na internetu zjistit ID daného bloku. Napovím, že ID tohoto bloku (ruditový blok) je 152.

Zde je možné si stáhnout svět s již připraveným bludištěm:
https://education.minecraft.net/lessons/programovani-agenta-bludiste/

Řešení:



Řešení v Javascriptu:

```
player.onChat("bludiste", function () {
    while (agent.inspect(AgentInspection.Block, DOWN) !=
 152) {
        if (agent.detect(AgentDetection.Block, RIGHT) &&
 !(agent.detect(AgentDetection.Block, FORWARD))) {
            agent.move(FORWARD, 1)
        } else {
            if (agent.detect(AgentDetection.Block, RIGHT
) && agent.detect(AgentDetection.Block, FORWARD)) {
                agent.turn(LEFT_TURN)
                agent.move(FORWARD, 1)
            } else {
                agent.turn(RIGHT_TURN)
                agent.move(FORWARD, 1)
            }
```

```
        }
    }
    player.say("Jsem venku z bludiště")
})
```

Programování Stavitele: Pyramida

Na rozdíl od Agenta postaví Stavitel daný objekt okamžitě. Nejsou vidět jednotlivé kroky, které provádí Agent.

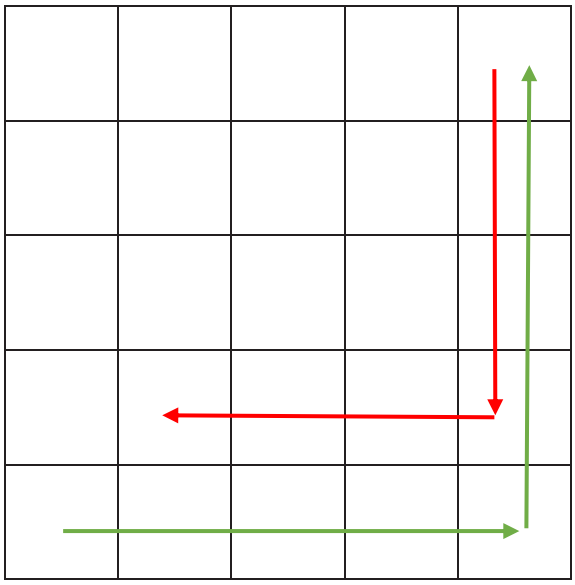
V této úloze postaví Stavitel pyramidu. Úlohu můžeme rozdělit na dvě úrovně.

1) Pyramida bude mít pevně danou délku základny a výšku na základě tohoto údaje dopočítáme.
2) Hráč si zvolí velikost základny sám a zbytek údajů se opět dopočítá.

V úloze 2 nahradíme konstantní číselné údaje proměnnými a budeme volat funkci s parametrem.

Úloha 1

Pyramida se postaví v místě, kde stojíme. Teleportujeme tak Stavitele na naši pozici. Umístíme si zde značku a necháme stavitele dojít po obvodu do protilehlého rohu spodního patra pyramidy (viz zelená čára). Dalším příkazem vyplníme tuto oblast libovolnými bloky. V následujícím kroku posuneme stavitele o patro výš a vrátíme se po stejné cestě zpět. Rozdíl bude pouze v tom, že při každé stěně ubereme o jeden krok, aby další patro bylo na každé straně o blok menší (viz červená čára). Délka nového patra bude oproti spodnímu o 2 bloky menší. Nyní můžeme celý postup zopakovat. Opakujeme tak dlouho, dokud nedocílíme vrcholu pyramidy.



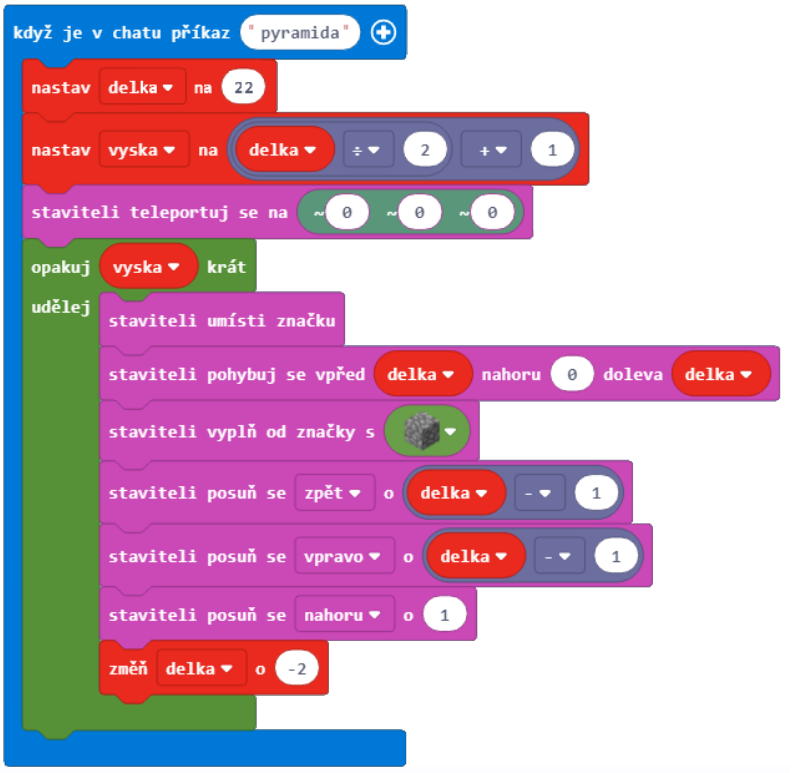
Výška:

Je nutné znát počet pater při zadané velikosti základny. Tento údaj se dá dopočítat, pokud dokážeme určit vzorec pro n-tý člen posloupnosti. Pokud žáci nemají znalost posloupností, dá se odhalit stylem pokus-omyl.

Např. Pyramida o délce základny 23 má patra o velikostech (23, 21, 19, 17, 15, 13, 11, 9, 7, 5, 3, 1). Postupem času se dá odhalit vzorec pro výšku pyramidy, který je:

$$výška = (délka\ základny\ \div 2) + 1$$

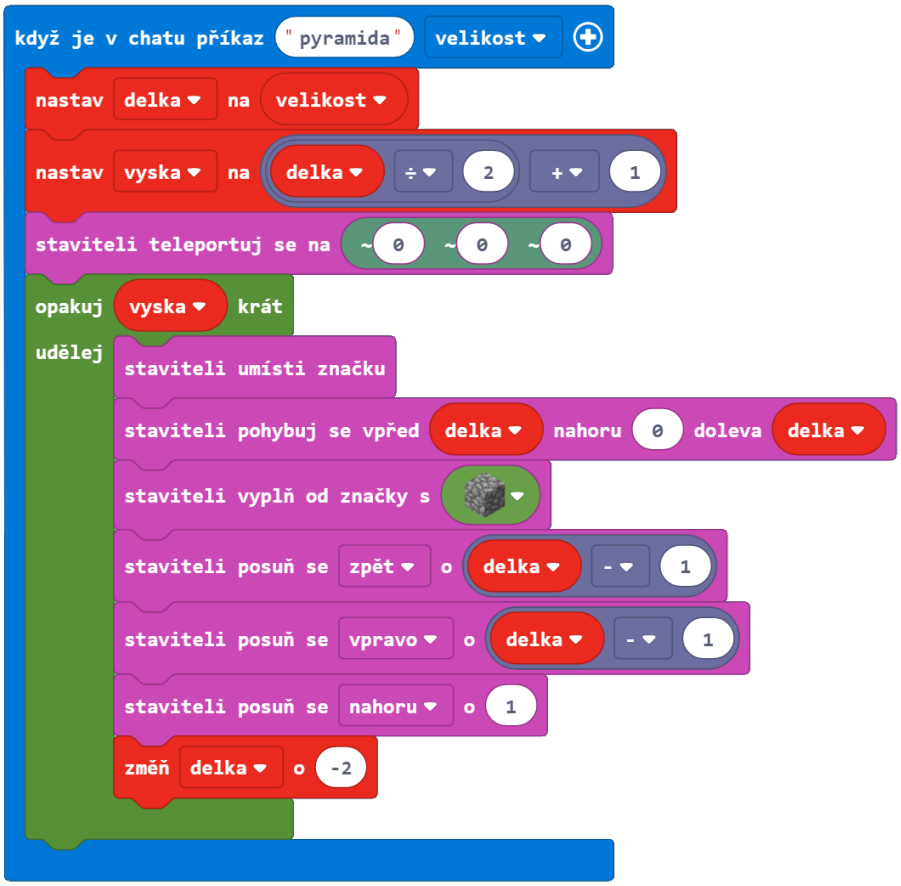
Řešení:



V Minecraftu nyní pouze napíše do chatovacího okna pyramida a postaví se pyramida o velikosti základny 23.

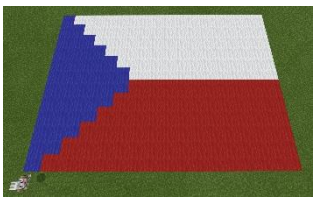
Úloha 2

Pouze rozšíříme tento kód o volání funkce s parametrem. Klikneme za slovem pyramida na + a zvolíme název parametru (např. velikost). Proměnnou „delka“ v úvodu programu nastavíme na velikost.



V Minecraftu nyní pouze napíšeme do chatovacího okna „pyramida“ a velikost základny (např. pyramida 46) a postaví se pyramida o velikosti základny 47 bloků.

Programování Agenta: Vlajka České republiky



Zadání:

Naprogramujte Agenta, aby položil na zem vlajku České republiky.

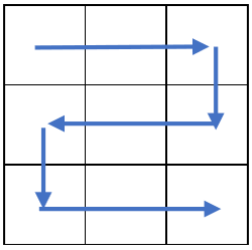
Řešení:

Celý kód naleznete ke stažení pod tímto odkazem: https://makecode.com/_iueHK4aDVcd5

Nejprve si úlohu načrtneme ideálně na čtverečkovaný papír. Určíme výšku a šířku celé vlajky. Modrý cíp vytvoříme tak, že v každém řádku vždy přidáme o jeden blok modré navíc. Pokud se pozorně podíváme na daný problém, zjistíme, že vlajka je až na rozdíl barev symetrická.

V našem případě bude mít vlajka výšku 12 bloků a šířku 17 bloků. Vlajku budeme vytvářet ze dvou polovin vždy 6 bloků vysokých. Agent bude pokládat bloky pod sebe, začne v levém horním rohu a bude pokračovat po řadách (viz Obrázek 2: Systém pohybu a otoček).

Na začátku navíc musíme do Agentova inventáře přidat tři typy bloků. Do slotu 1 modrý blok (modrá vlna), do slotu 2 bílý blok (bílá vlna) a do slotu 3 červený blok (červená vlna).



Obrázek 2: Systém pohybu a otoček

Postupně si vytvoříme 3 funkce.

První funkce „otocka“ bude pouze pomocná a zajistí automatické otáčení Agenta vždy, když narazí na konec řady (viz Obrázek 3: Algoritmus otáčení). Vytvoříme proměnnou *otocka2*. Když bude mít proměnná *otocka2* hodnotu Pravda, zahne agent doprava, v opačném případě doleva. Abychom zajistili pravidelné střídání směrů, změníme na konci programu po provedení otočky hodnotu proměnné *otocka2* na opačnou.

Obrázek 3: Algoritmus otáčení

Další funkce zajistí vytvoření první poloviny vlajky s modrobílým pozadím (viz Obrázek 5: Funkce pro obě poloviny vlajky). Funkci nazveme „bila“. Jako první nastavíme proměnnou *otocka2* na Pravda, abychom zajistili první otočku doprava. Dále si vytvoříme dvě nové proměnné. *Sirka* pro šířku vlajky a *modra* pro počet modrých kostek v daném řádku. Nastavíme proměnnou *sirka* na 17 a *modra* na 1, jelikož v prvním řádku je pouze jedna modrá kostka.

Algoritmus se bude opakovat od chvíle, kdy se Agent vrátí do stejné pozice jako na začátku. To je vždy, když dojde na jednu stranu a zpět, tj. postaví dva řádky.

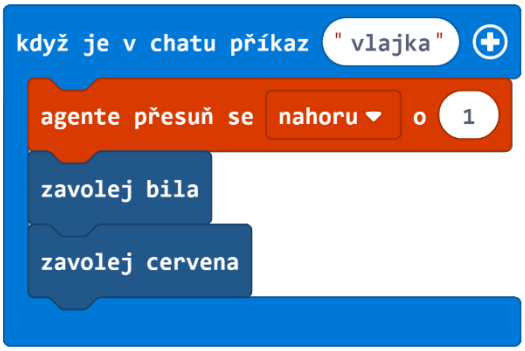
Nastavíme aktivní slot na 1 (modrá vlna). Opakujeme pokládání kostek tolikrát, kolik je nastavené v proměnné blue. V prvním řádku tedy 1x. Poté změníme aktivní slot na 2 (bílá vlna) a budeme kostky pokládat tolikrát, kolik zbývá do celé šířky vlajky. Počet bílých kostek spočítáme jako rozdíl šířky mínus počet modrých kostek v řádku. Tím vytvoříme jeden řádek vlajky. Změníme hodnotu modra o 1 a začneme vytvářet druhý řádek, tentokráte odzadu.

Zavoláme funkci otocka, aby se Agent otočil. Budeme opakovat celý postup s tím rozdílem, že nejprve začneme pokládat bílé bloky a až poté modré. Na konci opět zvýšíme hodnotu proměnné o 1.

Zopakujeme-li tento postup 3x dostaneme jednu polovinu celé vlajky.

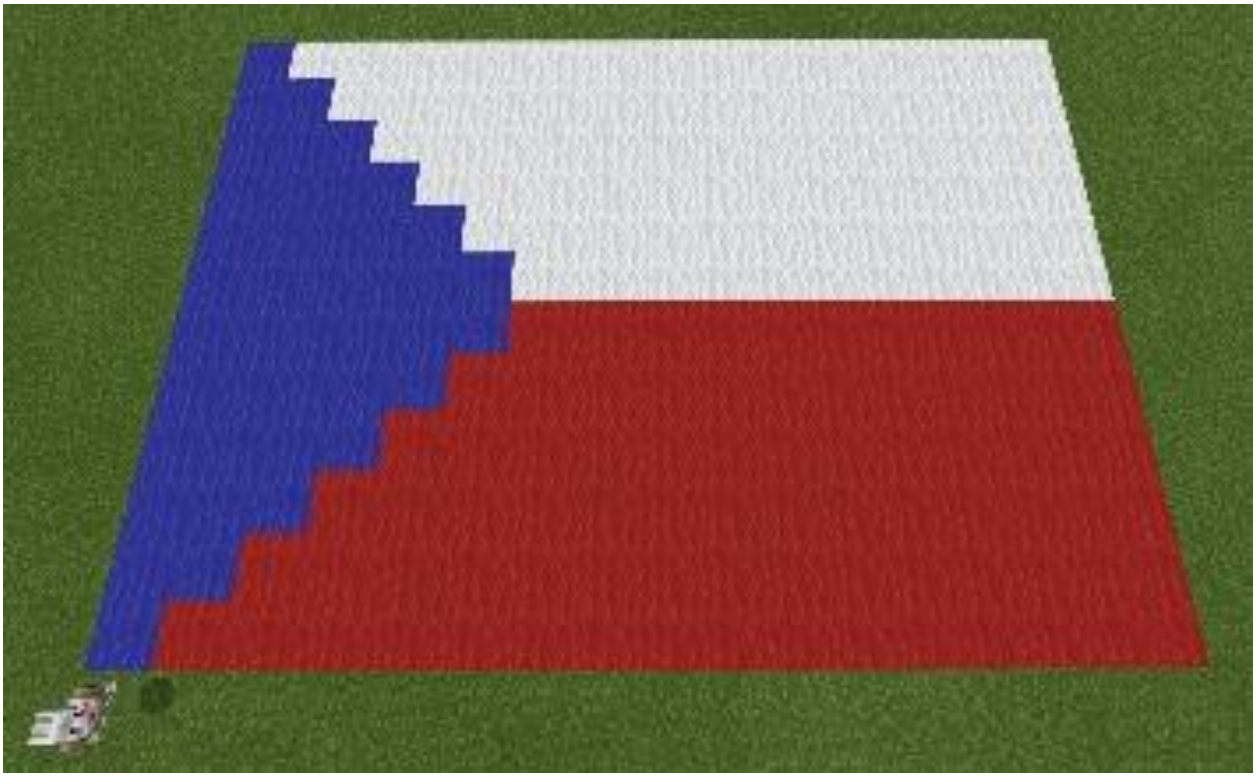
Druhá polovina vlajky je velmi obdobná s tím rozdílem, že tentokrát začínáme s 6 modrými kostkami a postupně v každém řádku jednu ubíráme. Místo bílých bloků pokládáme bloky červené. Vytvoříme si funkci „cervena“ zkopírováním funkce „bila“ a upravením patřičných bloků.

Na závěr už jen stačí vytvořit hlavní program, který v Minecraftu zavoláme v chatovacím okně napsáním názvu programu „vlajka“. Tento hlavní program nejprve posune Agenta o jeden blok výš, aby mohl pokládat další bloky pod sebe, a následně zavolá obě funkce „bila“ a „cervena“.



Obrázek 4: Hlavní program

Obrázek 5: Funkce pro obě poloviny vlajky



Obrázek 6: Výsledná vlajka České republiky

Řešení v Javascriptu:

```
let otocka2 = false
let sirka = 0
let modra = 0
function otocka () {
    if (otocka2) {
        agent.turn(RIGHT_TURN)
        agent.move(FORWARD, 1)
        agent.turn(RIGHT_TURN)
        agent.move(FORWARD, 1)
    } else {
        agent.turn(LEFT_TURN)
        agent.move(FORWARD, 1)
        agent.turn(LEFT_TURN)
        agent.move(FORWARD, 1)
        otocka2 = !(otocka2)
```

```
    }
}
player.onChat("vlajka", function () {
    agent.move(UP, 1)
    bila()
    cervena()
})
function bila () {
    otocka2 = true
    sirka = 17
    modra = 6
    for (let index = 0; index < 3; index++) {
        agent.setSlot(1)
        for (let index = 0; index < modra; index++) {
            agent.place(DOWN)
            agent.move(FORWARD, 1)
        }
        agent.setSlot(3)
        for (let index = 0; index < sirka -
 modra; index++) {
            agent.place(DOWN)
            agent.move(FORWARD, 1)
        }
        modra += -1
        otocka()
        for (let index = 0; index < sirka -
 modra; index++) {
            agent.place(DOWN)
            agent.move(FORWARD, 1)
        }
        agent.setSlot(1)
```

```
        for (let index = 0; index < modra; index++) {
            agent.place(DOWN)
            agent.move(FORWARD, 1)
        }
        otocka()
        modra += -1
    }
}
function cervena () {
    otocka2 = true
    sirka = 17
    modra = 1
    for (let index = 0; index < 3; index++) {
        agent.setSlot(1)
        for (let index = 0; index < modra; index++) {
            agent.place(DOWN)
            agent.move(FORWARD, 1)
        }
        agent.setSlot(2)
        for (let index = 0; index < sirka -
 modra; index++) {
            agent.place(DOWN)
            agent.move(FORWARD, 1)
        }
        modra += 1
        otocka()
        for (let index = 0; index < sirka -
 modra; index++) {
            agent.place(DOWN)
            agent.move(FORWARD, 1)
        }
```

```
        agent.setSlot(1)
        for (let index = 0; index < modra; index++) {
            agent.place(DOWN)
            agent.move(FORWARD, 1)
        }
        otocka()
        modra += 1
    }
}
```

Programování Agenta: Nejhlubší místo

Zadání:

Naprogramujte Agenta, aby zjistil v přesně dané čtvercové oblasti hloubku. Hráč teleportuje agenta těsně nad vodní hladinu a nechá Agenta prozkoumat hráčem zvolenou čtvercovou oblast. Agent po prohledání oblasti řekne, jaká je maximální hloubka dané vodní plochy.

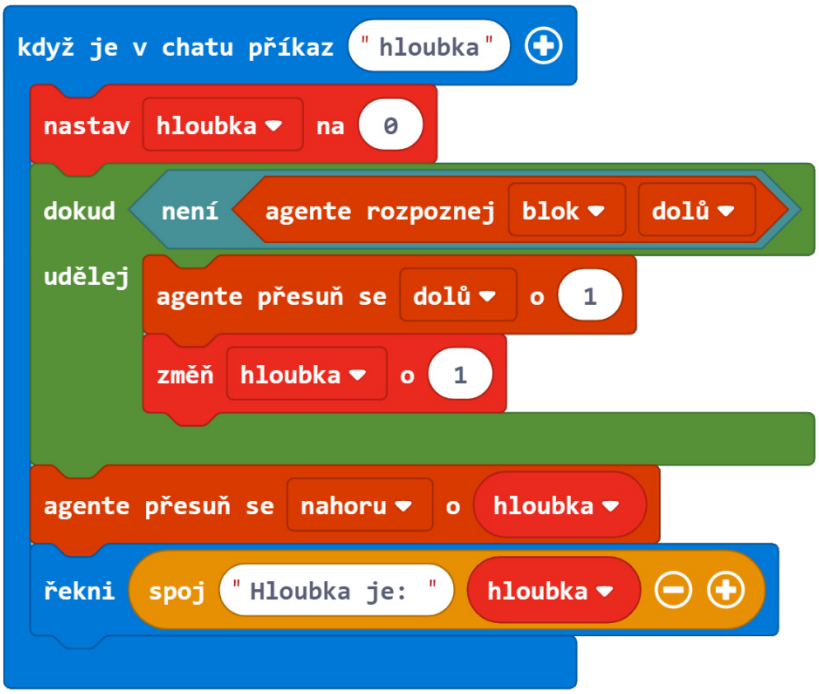
Řešení:

Úlohu rozdělíme do několika podúloh. Podle množství rozšíření můžete nastavit i obtížnost. V mém řešení se Agent vždy vrátí nad vodní hladinu. Jinou možností by mohlo být kopírovat dno. Takový algoritmus by mohl být efektivnější.

1) Agent zjistí hloubku v jednom místě jezera.
2) Agent zjistí hloubku v x za sebou jdoucích blocích vodní hladiny a v každém bodě vypíše, jaká je zde hloubka.
3) Rozšíříme úlohu dva o maximální hloubku. Agent na konci své cesty řekne, jaká byla maximální hloubka.
4) Agent prohledá pevně danou čtvercovou oblast. Velikost bude zadána jako konstantní číslo.
5) Rozšíříme úlohu 4 o možnost zvolit si velikost prohledávané plochy.

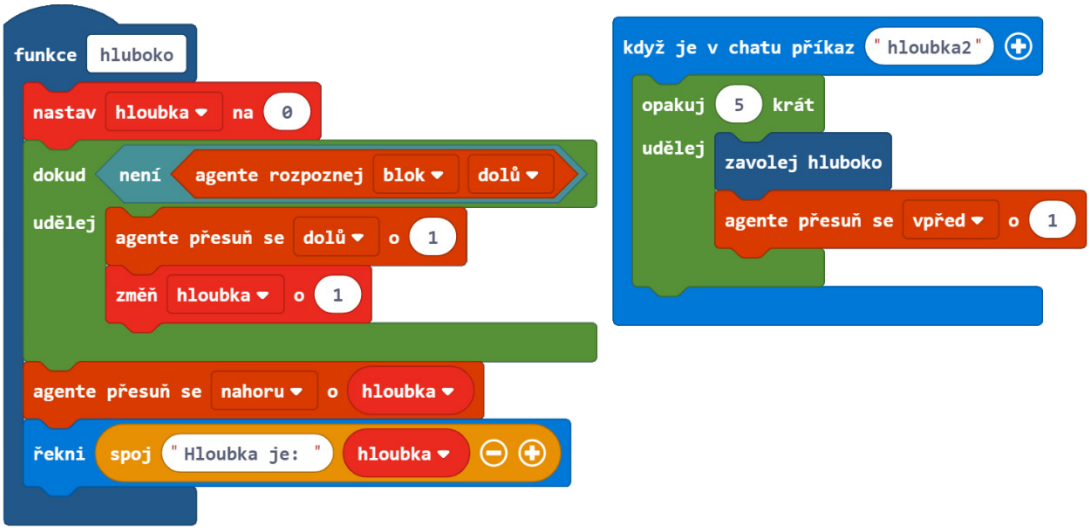
Úloha 1

Agent stojí nad vodní hladinou. V každém kroku zjišťuje, zda má pod sebou blok. Pokud ne, ponoří se o blok níž a zvýší hodnotu proměnné hloubka o 1. Nakonec se musí vynořit zpět nad hladinu.



Úloha 2

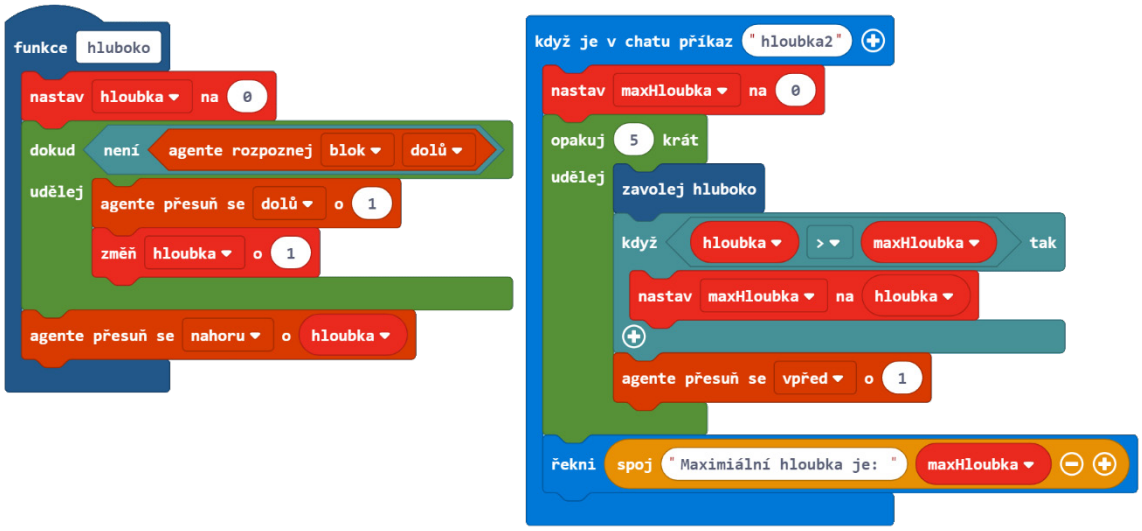
Použijeme kód z předchozí úlohy. Uložíme si ho jako funkci, kterou budeme v jednotlivých krocích volat. Funkci nazveme *hluboko*. Agent nyní projde část 5 bloků dlouhou a v každém kroku vypíše hloubku.



Úloha 3

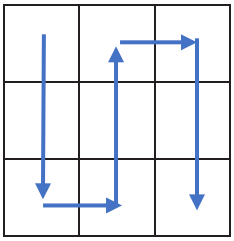
Rozšíříme kód v příkazu hloubka2 o uložení maximální hloubky. Vytvoříme si proměnnou maxHloubka a v každém kroku budeme porovnávat, zda je nová hloubka větší než

maxHloubka. V případě, že ano, přepíšeme hodnotu proměnné maxHloubka. Na závěr vypíšeme hodnotu nejhlubšího místa.



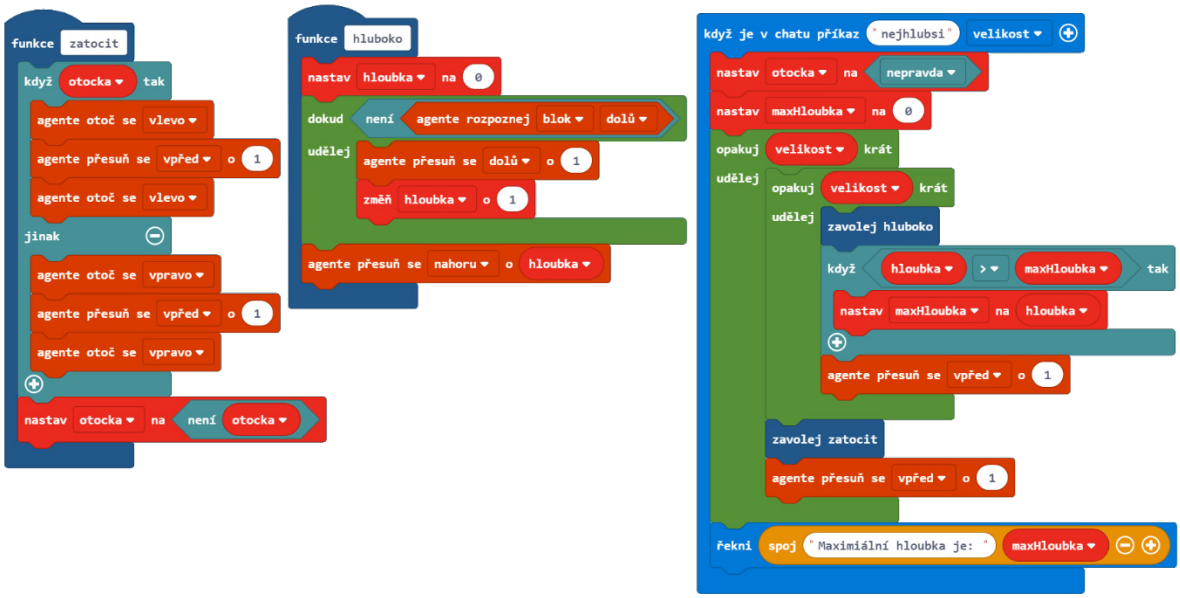
Úloha 4, 5

Naučíme Agenta prohledávat čtvercovou oblast. Budeme si muset vytvořit program pro otáčení se na konci každé řady. Tuto část kódu si opět uložíme do samostatné funkce *zatocit.* Vytvoříme si proměnnou otocka. Budeme střídat její hodnotu mezi true a false a stejně tak se budou střídat otočky na konci řady (jednou vlevo, jednou vpravo).

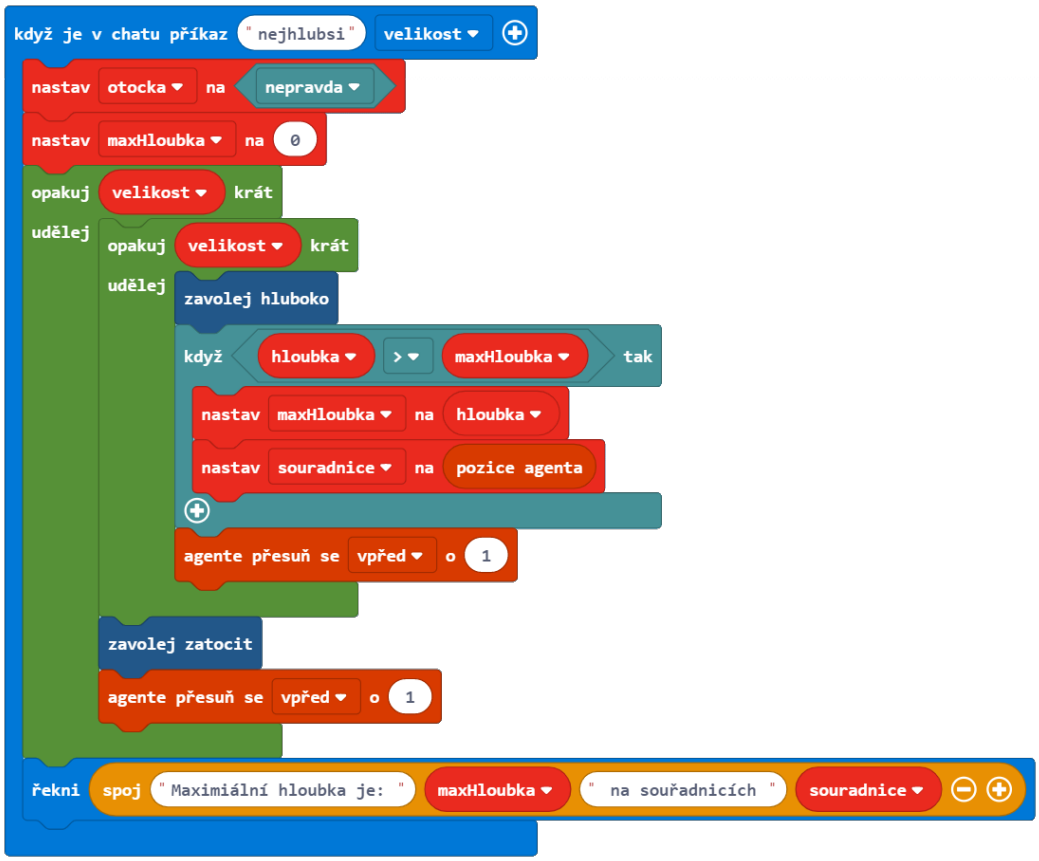


Rozšíříme program hloubka3. Nově ho nazveme *nejhlubsi.* Nastavíme otočku na false. Tím docílíme toho, že první otočka bude vpravo. Zakomponujeme otočku do kódu.

Posledním rozšířením je přidání parametru velikost do příkazu nejhlubsi. Hráč si tak bude moct zvolit velikost prohledávané oblasti. Do obou cyklů repeat ještě vložíme místo konkrétního čísla opakování proměnnou velikost a celý příkaz je hotový.



Posledním vylepšením může být určení nejhlubšího místa. Nemusí nás přeci jen zajímat, jak hluboké je nejhlubší místo, ale i na jakých souřadnicích se nachází. Vytvoříme si novou proměnnou s názvem *souradnice*. Vždy v nejhlubším naměřeném místě (tj. v podmínce když) uložíme aktuální souřadnice agenta do této proměnné. Na závěr v bloku řekni už tyto souřadnice pouze vypíšeme.



V Minecraft: Education Edition napíšeme do chatovacího okna příkaz „nejhlubsi 5“. Agent prohledá pole o velikosti 5x5 a vypíše maximální hloubku této oblasti a souřadnice nejhlubšího místa.

Řešení v Javascriptu:

```
let otocka = false
let hloubka = 0
let maxHloubka = 0
let souradnice: Position = null
function zatocit () {
    if (otocka) {
        agent.turn(LEFT_TURN)
        agent.move(FORWARD, 1)
        agent.turn(LEFT_TURN)
    } else {
        agent.turn(RIGHT_TURN)
        agent.move(FORWARD, 1)
        agent.turn(RIGHT_TURN)
    }
    otocka = !(otocka)
}
function hluboko () {
    hloubka = 0
    while (!(agent.detect(AgentDetection.Block, DOWN)))
{
        agent.move(DOWN, 1)
        hloubka += 1
    }
    agent.move(UP, hloubka)
}
player.onChat("nejhlubsi", function (velikost) {
    otocka = false
    maxHloubka = 0
    for (let index = 0; index < velikost; index++) {
        for (let index = 0; index < velikost; index++) {
```

```
            hluboko()
            if (hloubka > maxHloubka) {
                maxHloubka = hloubka
                souradnice = agent.getPosition()
            }
            agent.move(FORWARD, 1)
        }
        zatocit()
        agent.move(FORWARD, 1)
    }
    player.say("Maximiální hloubka je: " + maxHloubka +
" na souřadnicích " + souradnice)
})
```

education.minecraft.net

Microsoft

Publikace obsahuje popisy, názvy funkcí a snímky obrazovek pořízené v únoru 2020.
V době, kdy čtete tuto publikaci, se může aktuální prostředí aplikace Minecraft: Education Edition a MakeCode lišit.

aka.ms/skolstvi