Úvod do Unity ML-Agents

Využití umělé inteligence v Unity Game Engine

Tento manuál pracuje s *ML-Agents release 18*, tudíž u novějších verzí nemusí dané postupy již nadále platit.

Obsahuje přesnou přípravu projektu pro práci s ML-Agents, a obsahuje taktéž i velmi jednoduchý příklad.

Celý manuál počítá s tím, že uživatel je již seznámen s vývojovým prostředím Unity Engine, a tudíž zde nebudou popsány postupy, jak vytvářet projekty, či importovat balíčky. Dále tento manuál předpokládá, že uživatel má již daný Engine na svém zařízení nainstalovaný, a plně funkční.

Obsah

[2. Co jsou ML-Agents 3](#_Toc89794743)

[3. Manuál přípravy trénovacího prostředí 3](#_Toc89794744)

[**Krok 1 - Instalace Pythonu 3**](#_Toc89794745)

[**Krok 2 - Kontrola úspěchu instalace 3**](#_Toc89794746)

[**Krok 3 - Aktualizace PIPu 3**](#_Toc89794747)

[**Krok 4 - Vytvoření prostředí 3**](#_Toc89794748)

[**Krok 5 - Aktivace prostředí 4**](#_Toc89794749)

[**Krok 6 - Instalace balíčků 5**](#_Toc89794750)

[**Krok 7 - Ověření funkčnosti 5**](#_Toc89794751)

[4. Manuál přípravy Unity projektu 5](#_Toc89794752)

[**Krok 1 - Vytvoření projektu 5**](#_Toc89794753)

[**Krok 2 - Povolení Preview Balíčků 6**](#_Toc89794754)

[**Krok 3 - Importování ML-Agents balíčku 6**](#_Toc89794755)

[5. Příklad použití ML-Agents 7](#_Toc89794756)

[6. Odkazy 10](#_Toc89794757)

# Co jsou ML-Agents

Je open-source projekt, který umožňuje vývojářům využívat a trénovat umělou inteligenci v Unity Game Engine. Jedná se nyní o nejpřístupnější a nejpokročilejší řešení pro toto prostředí.

# Manuál přípravy trénovacího prostředí

V tomto kroku bude popsána příprava trénovací prostředí pro ML-Agents. Toto prostředí se poté napojí na Unity, a bude využito pro učení agentů umělé inteligence.

## Instalace Pythonu

ML-Agents vyžaduje pro svoji funkčnost *Python*, konkrétně verzi *3.6/3.7*. V postupu se ukázalo jako funkční verze *3.7*, jejíž installer se dá v době psaní stáhnout na [oficiálních stránkách Pythonu](https://www.python.org/downloads/release/python-370/).

## Kontrola úspěchu instalace

Po instalaci otevřete Command Prompt (příkazový řádek, dále CMD), a vypište *python*, či *py* (pokud *python* nebude fungovat). Pokud instalace proběhla úspěšně, objeví se verze nainstalovaného pythonu.  
Následovně vypište *exit()*, pro zrušení python módu.

Obsah obrázku text

Popis byl vytvořen automaticky

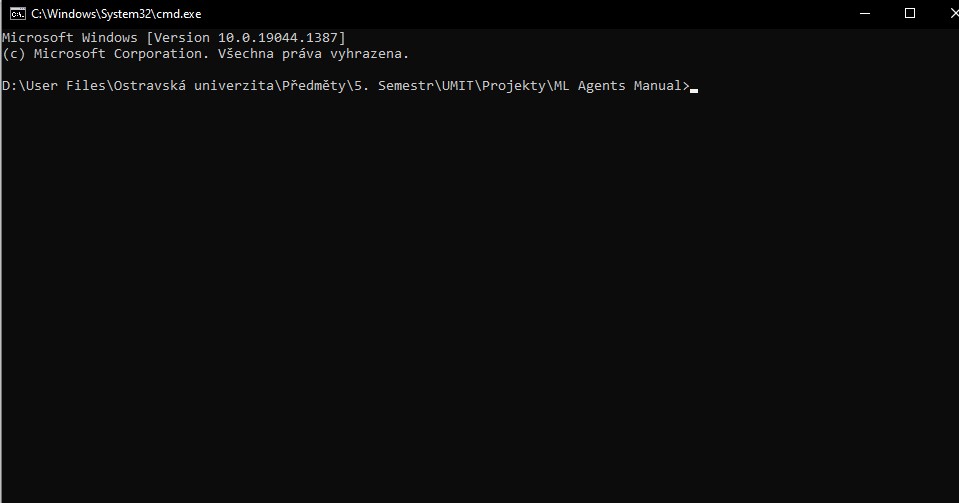
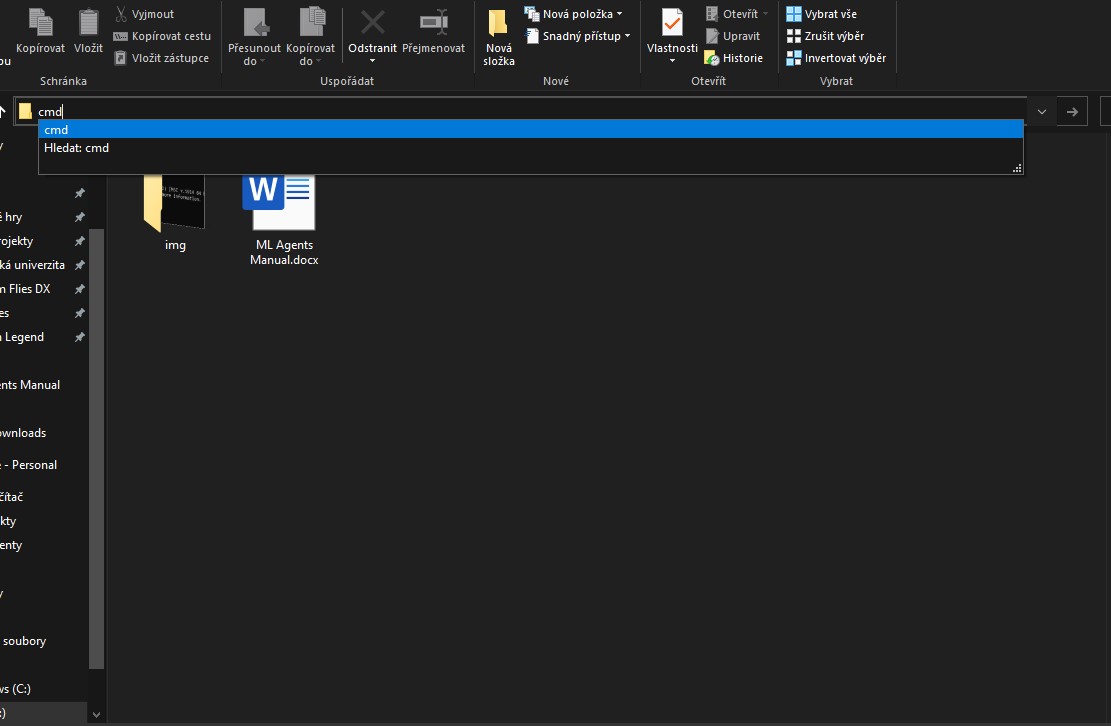
## Aktualizace PIPu

PIP (Package Installer for Python) je program obstarávající instalaci různých balíčků pro Python. Bude využit pro stažení ML-Agents a PyTorche, tudíž, pro zmenšení možnosti selhání, by měl být aktualizovaný na nejnovější verzi. Do CMD zadejte komand:

*python -m pip install –upgrade pip*

## Vytvoření prostředí

Nyní je třeba vytvořit trénovací prostředí. Toto prostředí bude vyžadovat aspoň *4,5 GB* místa. Přenavigujte CMD do složky, ve které jej chcete vytvořit. (Tohle může být uděláno komandem  
*cd CESTA* nebo otevřením složky v prohlížeči, a do adresy zadat *cmd*)



Pro vytvoření prostředí zadejte komand:

*python -m venv JMÉNO\_PROSTŘEDÍ*

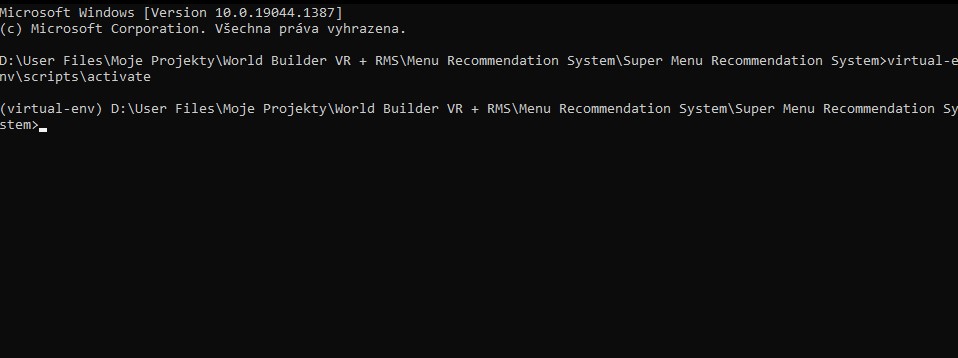
Tím se vytvoří složka, obsahující nutné soubory.  
Pozor! Prostředí za žádnou cenu nevytvářet pod adresou, obsahující jiné znaky než anglické.  
Prostředí se potom nebude chtít napojit. Například cesta v příkladě nahoře nebude validní, protože obsahuje znaky „á“, „ř“, atd.

## Aktivace prostředí

Pokaždé, když budete chtít pracovat s daným prostředím, je nutno jej aktivovat komandem:

*NÁZEV\_PROSTŘEDÍ\scripts\activate*

Tím by se mělo prostředí aktivovat.



## Instalace balíčků

Nyní je třeba do daného prostředí nainstalovat PyTorch a ML-Agent balíčky. Zadejte komadny:

*pip3 install torch~=1.7.1 -f https://download.pytorch.org/whl/torch\_stable.html*

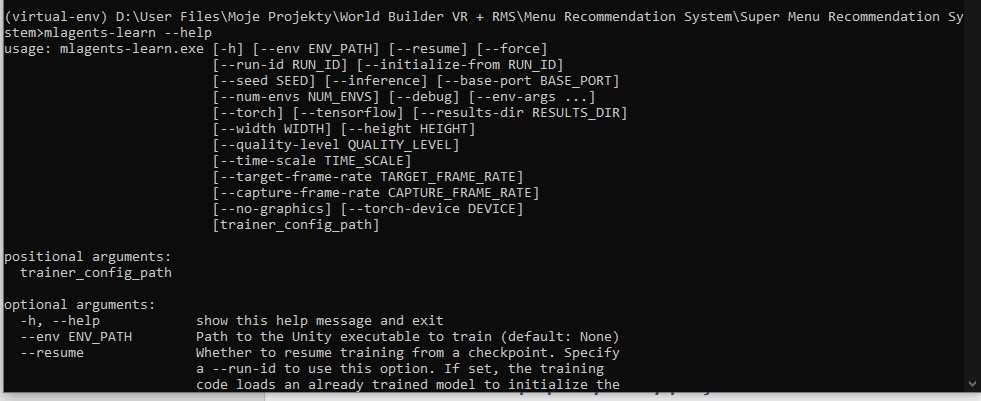
*python -m pip install mlagents==0.27.0*

pro instalaci PyTorche a ML-Agents respektivně.

## Ověření funkčnosti

Prostředí by mělo být připravené. Ověřte, že vše funguje zadáním komandu:

*mlagents-learn --help*



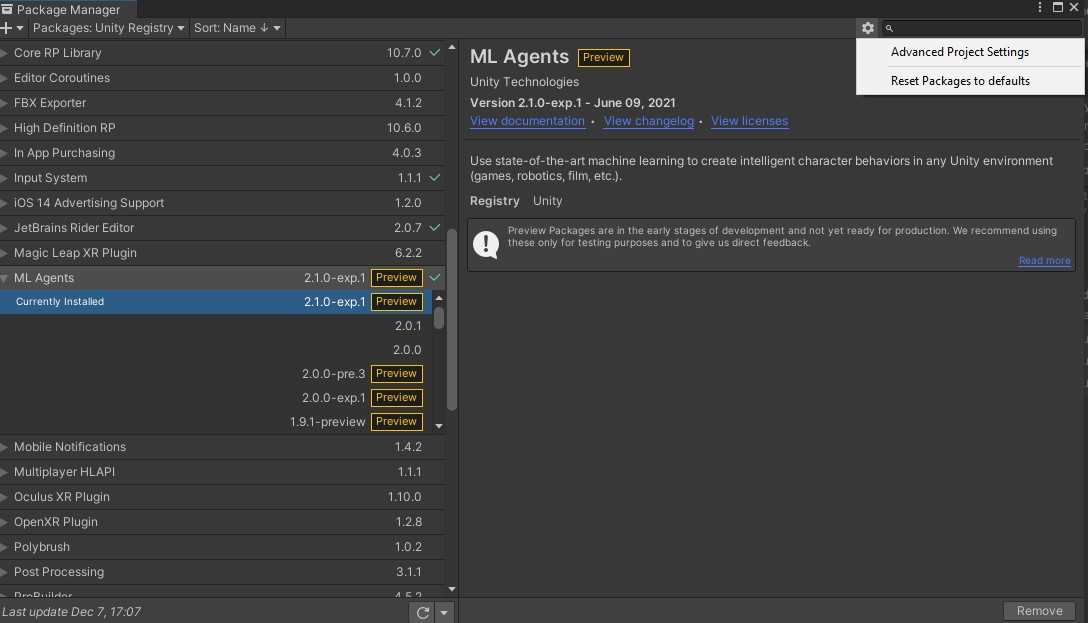
# Manuál přípravy Unity projektu

## Vytvoření projektu

Otevřete Unity Hub, a vytvořte nový projekt.

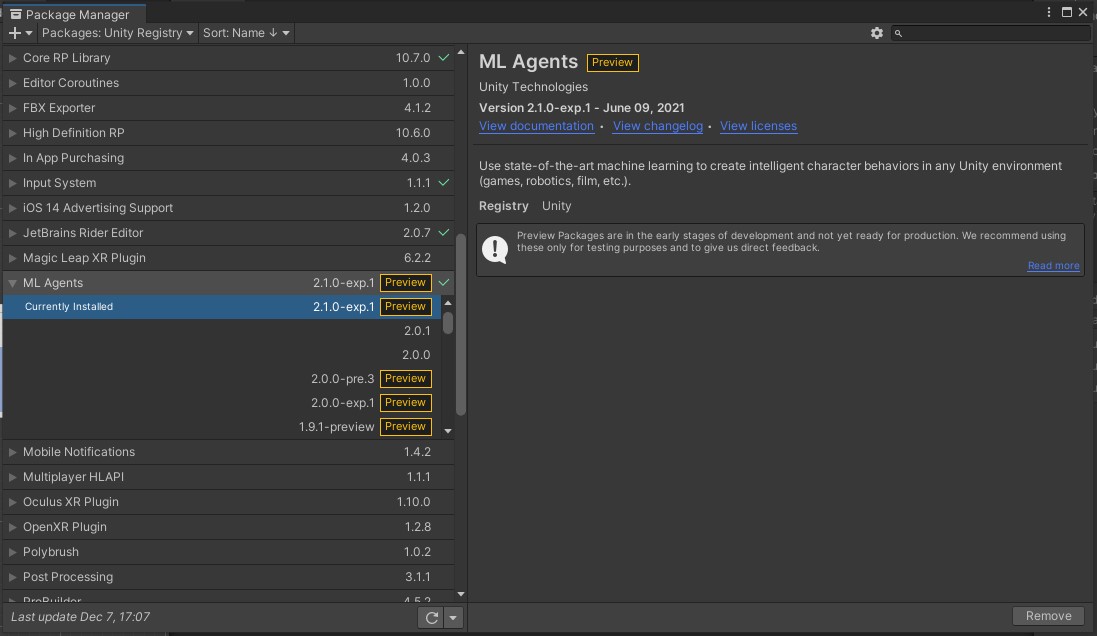
## Povolení Preview Balíčků

Otevřete *Package Manager*, zvolte *Advanced Package Settings*, a zatrhněte *Enable Preview Packages* v Advanced Settings.



## Importování ML-Agents balíčku

Otevřete Package Manager, přejděte do *Unity Registry* a importujte *ML Agents 2.1.0*, popřípadě pokud tak vysoká tam není, tak nejnovější verzi.

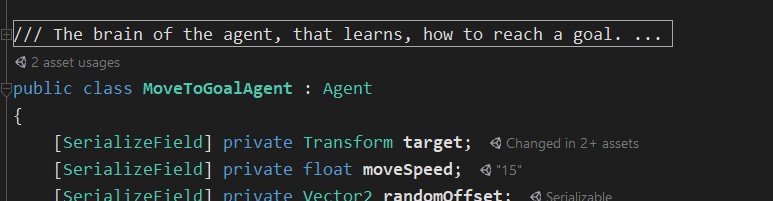


Projekt by nyní měl být připraven na použití.

# Příklad použití ML-Agents

K manuálu je přibalený příklad užití ML-Agents. Daný skript stačí hodit na nějaký objekt, a stane se z něj agent, snažící se dotknout cílového objektu.

Všechny třídy, které mají fungovat jako Ml Agent musí dědit třídu Agent. Tam jim umožní provádět všechny potřebné akce, které agent musí provádět.



Každý script dědící Agent musí zároveň využívat komponent *Behavior Parameters*. Zde se nastaví, kolik údajů bude agent sledovat, počet akcí, které bude provádět, a další nastavení.

Důležitá nastavení

* **Vector Observation – Space Size**

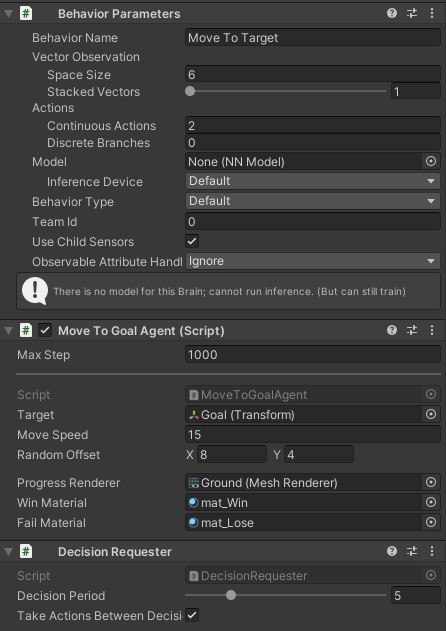
Nastaví, kolik údajů bude agent pozorovat. Ty se potom nastaví v metodě CollectObservations(). Musí být číselné hodnoty.

* **Continuous Actions**

Nastaví, kolik akcí agent bude provádět. Continuous znamená, že výstupní čísla budou desetinná. Pro celá čísla je nutno navýšit Discrete Branches, a tam nastavit Discrete Actions.

* **Behavior Type**

Zvolit, zda se agent bude učit (Default), bude ovládán manuálně (Heuristic), či bude řízen předučeným modelem (Inference).



Pro Agenta jsou nejdůležitější 4 metody

* **OnEpisodeBegin()**

MLAgents funguje na principu epizod. Nové epizody vycházejí ze starých, čímž se agent učí. V této metodě se řekne, co se má stát na začátku každé epizody. (V daném příkladě se přesune agent a cíl na nějakou náhodnou pozici.) Epizody se končí manuálně metodou EndEpisode().

* **CollectObservations()**

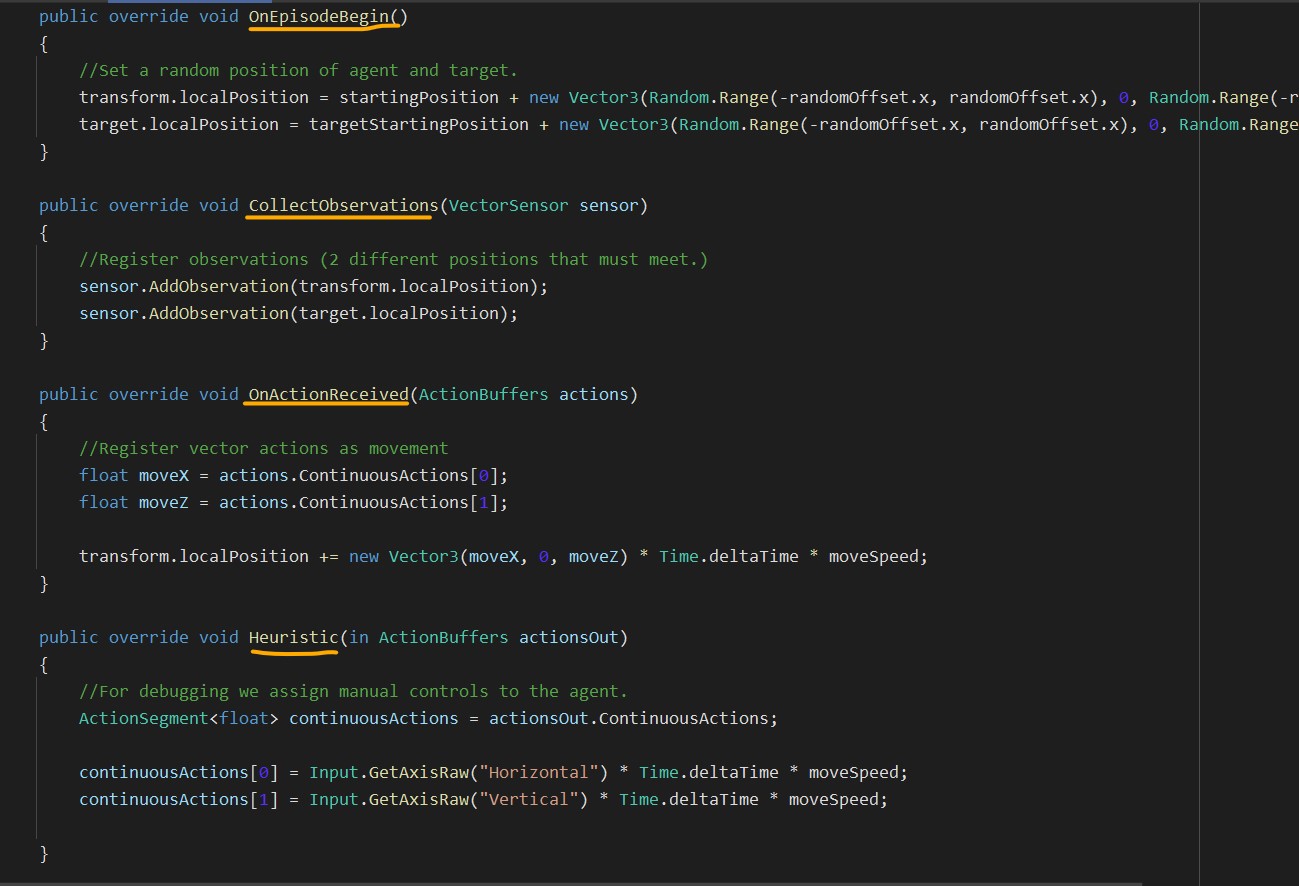
Tato metoda je využita jako pozorování relevantních (a pouze číselných) údajů pro agenta. Pozoruje to, co nějak ovlivní jeho akce. (V příkladu agent pozoruje svojí pozici a pozici svého cíle, tudíž jakožto Vectory3 celkem 6 hodnot.)

* **OnActionsReceived()**

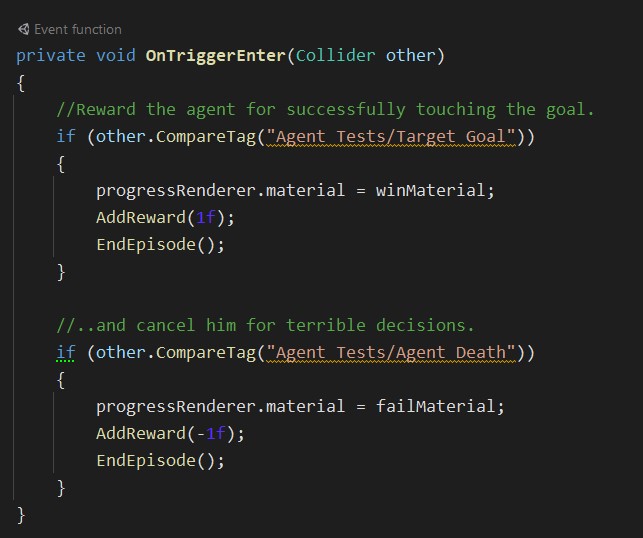
Zde se nachází logika, kterou agent provádí v každý moment, kdy je vyvoláno rozhodnutí činit úkon. (Rozhodnutí je většinou voláno manuálně metodu RequestDecision() nebo Unity Componentem DecisionRequestem, který volá DecisionRequest v nějakém intervalu. V příkladu agent mění svojí pozici na základě vypočítaných údajů.)

* **Heuristic()**

V momentě, kdy zrovna není program napojený na testovací prostředí, či potřebuje uživatel otestovat svoji logiku se hodí tato metoda. Funguje jako manuální činění nastaveného úkonu. (V příkladě jsou 2 vstupy nastaveny jako stisknutí klávesových šipek.)



Co rozhoduje mezi úspěšným učením a selháním agenta je systém přídělů odměn. Voláním metody *AddReward()* je možné agentovi říct, zdali jeho akce jsou přijatelné, či nikoliv. V příkladě agent dostává odměnu za to, že se dostal k cílí, a ztrácí ji, když vrazí do stěn.



# Odkazy

ML-Agents na GitHubu: <https://github.com/Unity-Technologies/ml-agents>

Instalační návod od Unity: <https://github.com/Unity-Technologies/ml-agents/blob/main/docs/Installation.md>

Instalace a stejný příklad na Youtube: <https://www.youtube.com/watch?v=zPFU30tbyKs&list=PLzDRvYVwl53vehwiN_odYJkPBzcqFw110&index=2&t=332s>

Python 3.7 Instalátor: <https://www.python.org/downloads/release/python-370/>