**Plánujeme hru**

Začít programovat cokoliv, než víme, co přesně chceme, není úplně dobrý nápad. Proto, než začneme, bylo by dobré si chvíli sednout, připravit si šálek kávy či čaje a promyslet, co je naším cílem. Odborně se této fázi procesu vývoje software říká **analýza**.

Po vysoce osvěžujícím šálku teplého nápoje a chvíli přemýšlení můžeme dojít k následujícím požadavkům:

* na obrazovce budou nalevo a napravo obdélníky, které reprezentují hráče
* hráči se mohou pohybovat nahoru a dolů
* mezi nimi bude létat čtverec, představující míč
* míč bude postupem času zrychlovat
* pokud míč narazí do vrchní či spodní části obrazovky nebo hráče, odrazí se
* pokud míč narazí do levé nebo pravé části obrazovky, přičte bod protilehlému hráči
* skóre se zobrazuje nahoře uprostřed obrazovky
* (nastavitelné) druhý hráč může být ovládán počítačem

**Stavy hry**

Když tedy již zhruba tušíme, jak bude naše hra vypadat, nejspíše jsme došli k závěru, že naše hra bude mít celkem tři stavy:

1. **Začátek hry**, výběr, jestli chceme hrát s druhým hráčem, nebo proti počítači
2. **Začátek kola**, hra je pozastavena, aby se měli hráči čas nachystat
3. **Samotná hra**, která je ukončena až chybou jednoho z hráčů

Zároveň by také nebylo špatné, kdyby naše hra běžela pokaždé se stejným poměrem stran, nezávisle na velikosti obrazovky uživatele.

**Co je pro nás nového?**

Určitě jste si všimli, že pár věcí jsme se ještě neučili. Jedná se převážně o tato témata.

**Kreslení textu**

Pokud si chceme na obrazovku vykreslit text, musíme si nejdříve vytvořit písmo.

**Systémové písmo**

Pokud chceme použít **systémové písmo**, použijeme:



, kde font\_name je jméno nainstalovaného písma a size je poté velikost v pixelech.

Seznam všech dostupných systémových písem získáme pomocí pygame.font.get\_fonts().

**Vlastní písmo**

Pokud nechceme použít nějaké již nainstalované písmo, ale místo toho raději nějaké **vlastní písmo**, které sami přibalíme k našemu programu, nahrajeme jej pomocí:

Obsah obrázku text, Písmo, snímek obrazovky, Grafika

Obsah vygenerovaný umělou inteligencí může být nesprávný.

Parametr size zůstává stejný, ale místo jména systémového písma nyní vkládáme cestu k .ttf souboru s naším písmem. Pokud jako path vložíme None, použije se výchozí písmo.

**Vykreslení písma**

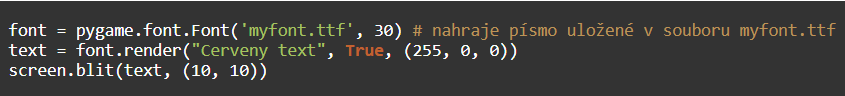
Jakmile máme takto vytvořené písmo, vyrenderujeme jej pomocí:

Obsah obrázku Písmo, text, Grafika, snímek obrazovky

Obsah vygenerovaný umělou inteligencí může být nesprávný.

, kde txt je string, který chceme vyrenderovat; True je parametr pro antialiasing, který zaoblí rohy písma *(což je obecně žádoucí)* a color je barva písma.

Nyní bychom měli v proměnné text uložený povrch, který nám ještě zbývá vykreslit na obrazovku. Takže pokud bychom chtěli vykreslit červený text na pozici (10, 10), mohl by náš kód vypadat nějak takto:

**Zachování velikosti při libovolné velikosti obrazovky**

Doteď jsme si vždy vytvořili proměnnou screen = pygame.display.set\_mode(...). Takto jsme museli nastavit pevné rozměry, nebo zjišťovat rozměry při vykreslení na celou obrazovku. Mnohem snadnější přístup by ale byl, pokud bychom mohli vykreslovat pouze pro jedno rozlišení a pak celou obrazovku automaticky převést na jiné. Tohoto efektu můžeme dosáhnout následovně:

Obsah obrázku text, snímek obrazovky, Písmo

Obsah vygenerovaný umělou inteligencí může být nesprávný.**Pohybování obdélníky**

Pro kreslení obdélníků jsme zadali souřadnice x a y, čímž jsme v každém cyklu vytvořili nový obdélník na nové pozici. Samotný [obdélník](https://www.pygame.org/docs/ref/rect.html) v pygame je ale mnohem mocnější. Pro začátek nám bude stačit znát funkci rect.move\_ip(dx, dy), která posune obdélníkem o dx pixelů doprava a o dy pixelů dolů. Existuje také funkce rect.move(dx, dy), která ale místo posunutí původního obdélníku vrátí nový obdélník s posunutím.

**Kolize obdélníků**

Obdélníky v pygame se za nás také postarají o zjištění kolizí, tedy zda se střetávají s nějakým jiným objektem. Hlavní funkce, které by nás mohly zajímat, jsou:

* rect.collidepoint((x, y)) - vrací True, pokud se bod (x, y) nachází uvnitř obdélníku
* rect.colliderect(rect2) - vrací True, pokud mají dva obdélníky průnik
* rect.collidelist(lst) - vrací True, pokud alespoň jeden z obdélníků uvnitř seznamu má průnik s hlavním
* rect.collidelistall(lst) - vrací True, pokud všechny z obdélníků uvnitř seznamu mají průnik s hlavním

https://www.itnetwork.cz/python/pygame/pygame-pong-priprava