1. Vytvořte si v nové složce s názvem **samostatná práce OOP** soubory **index.html** a **index.js**. V souboru **index.html** vytvořte základní kostru html dokumentu a zaveďte do něj odkaz na soubor **index.js**. Ve složce **samostatná práce OOP** si dále vytvořte podsložku s názvem **images** a zkopírujte si do ní všechny obrázky, které Vám byly dány k dispozici. Do hlavičky html dokumentu zkopírujte následující styl

```
<style>
        img{
                display: none;
        }
</style>
```

   A do těla dokumentu zkopírujte následující elementy

```
<img src="images/meteor.png" id="meteor">
<img src="images/cactus.png" id="cactus">
<img src="images/dinosaur-standing.png" id="player_standing">
<img src="images/dinosaur-crouching.png" id="player_crouching">
<img src="images/Layer_0.png" id="layer0">
<img src="images/Layer_1.png" id="layer1">
<img src="images/Layer_2.png" id="layer2">
<img src="images/Layer_3.png" id="layer3">
<img src="images/Layer_4.png" id="layer4">
<img src="images/background.png" id="background">
```

2. První třída, kterou budete vytvářet, bude pomocná třída pro reprezentaci dvojrozměrných vektorových proměnných. Tuto třídu budete používat pro ukládání pozice objektů ve scéně a pro ukládání jejich rychlosti pohybu.
    a. Vytvořte novou třídu (ideálně do samostatného souboru – nezapomeňte zavést odkaz na soubor do html dokumentu) s názvem **Vector**.

    b. Vytvořte konstruktor třídy s parametry **x** a **y**. Uvnitř konstruktoru přiřaďte hodnoty parametrů **x** a **y** atributům **this.x** a **this.y**.

    c. Dále vytvořte metodu **add**. Parametr této metody bude mít název **other**. Uvnitř metody proveďte přičtení hodnot atributů **x** a **y** instance **other** k hodnotám atributů **x** a **y** instance **this**.

    d. Poté vytvořte metodu **subtract**. Tato metoda bude stejná jako metoda **add**, akorát budete hodnoty atributů instance **other** od instance **this** odečítat.

    e. Vytvořte metodu sum s parametrem **other**. Uvnitř metody vraťte novou instanci třídy **Vector**. První argument konstruktoru vracené instance bude součet atributů **x** instance **this** a **other** a druhý bude součet atributů **y** instance **this** a **other**.

    f. Poté vytvořte metodu **difference**. Tato metoda bude stejná jako metoda **sum**, akorát místo součtu bude jako hodnoty argumentů nové instance třídy **Vector** používat rozdíl hodnot atributů instance **this** a **other**.

g. Vytvořte **getter** s názvem **squareSize**. **Getter** vrátí hodnotu **this.x * this.x + this.y * this.y**

3. Další třída, kterou budete vytvářet, je třída, která usnadní vykreslování.
   a. Vytvořte novou třídu s názvem **Scene**.

   b. Vytvořte konstruktor třídy s parametry **width**, **height**. Uvnitř konstruktoru přiřaďte hodnoty parametrů **width**, **height** atributům **this.width**, **this.height**.

      Následně uvnitř konstruktoru vytvořte plátno pomocí **document.createElement("canvas")**. Výstup z této metody uložte do lokální proměnné. Poté vytvořenému plátnu nastavte atribut **width** na **this.width** a atribut **height** na **this.height**.

      Plátno vložte pomocí metody **document.body.appendChild** do dokumentu.

      Poté uložte 2D kontext plátna do atributu **this.context**. (2D kontext plátna získáte pomocí metody **getContext("2d")**, kterou zavoláte přes proměnnou do které jste si vytvářené plátno ukládali)

   c. Vytvořte metodu **draw**, která bude přebírat 0 a více argumentů. Za tímto účelem použijte **spread operátor** (tři tečky před názvem parametru) a parametr nazvěte **drawables**.

      Uvnitř metody vykreslete všechny objekty v poli **drawables**. Vykreslení objektu uloženého na indexu 0 v poli **drawables** bude vypadat následovně: **drawables[0].draw(this.context)** (Ne, že u někoho uvidím, jak tam má tento řádek zkopírovaný. Jedná se pouze o příklad toho, jak budete objekty uvnitř pole vykreslovat. Nápověda – v hranatých závorkách nebude 0, ale něco jiného.)

4. Po vytvoření scény budete vytvářet třídu pro vykreslování pozadí.
   a. Vytvořte třídy s názvem **Background**.

   b. Konstruktor této třídy bude obsahovat parametry **width**, **height**.

   c. Hodnoty těchto parametrů uvnitř konstruktoru přiřaďte atributům **this.width**, **this.height**. Poté do atributu **this.image** uložte výstup z metody **document.getElementById("background")**

   d. Uvnitř této třídy vytvořte metodu **draw**, která bude mít jeden parametr s názvem **context**. Uvnitř této metody vykreslete pomocí parametru **context** obrázek uložený v atributu **this.image**, jehož levý horní roh bude na souřadnicích **0**, **0** a jeho velikost bude **this.width**, **this.height**. K vykreslení obrázku použijte metodu **drawImage**, kterou zavoláte přes parametr **context**. Argumenty metody budou **this.image**, **0**, **0**, **this.width**, **this.height**.

5. Vyzkoušejte, zda třídy **Scene** a **Background** fungují.

a. Uvnitř souboru index.js vytvořte instanci třídy **Scene** s názvem **scene** a jako argumenty konstruktoru zadejte Vámi zvolené hodnoty (například **1000**, **700**).

   Poté vytvořte instanci třídy **Background** s názvem **background** a jako argumenty konstruktoru použijte hodnoty **scene.width**, **scene.height**.

   Poté přes proměnnou **scene** zavolejte metodu **draw** a jako argument zadejte proměnnou **background**. Když spustíte aplikaci v prohlížeči, měli byste vidět obrázek pozadí.

6. Nyní budete vytvářet třídu, ze které budou dědit všechny třídy, kterým bude možné nastavovat pozici.
   a. Vytvořte třídu s názvem **Positionable**.

   b. Konstruktor této třídy bude obsahovat pouze jeden parametr s názvem **position**. Hodnotu tohoto parametru uložte do atributu **this.position**.

   c. Vytvořte gettry a settry s názvem **x** a **y**. Getter **x** bude vracet hodnotu **this.position.x**. Getter **y** bude vracet hodnotu **this.position.y**. Setter **x** bude mít parametr **value** jehož hodnotu přiřadí atributu **this.position.x**. Setter **y** bude mít parametr **value** jehož hodnotu přiřadí atributu **this.position.y**.

7. Dále vytvoříte třídu pro vyhodnocování kolizí.
   a. Vytvořte třídu s názvem **CollisionDetection**.
   b. Uvnitř této třídy vytvořte statickou metodu s názvem **checkRectCollision**, která bude mít dva parametry s názvy **rect1** a **rect2**. Uvnitř metody vraťte následující hodnotu

      (rect1.rightEdge > rect2.leftEdge) && (rect2.rightEdge > rect1.leftEdge)&&
      (rect1.bottomEdge > rect2.topEdge)&& (rect2.bottomEdge > rect1.topEdge)

   c. Vytvořte statickou metodu s názvem **checkCircleCollision**, která bude mít dva parametry **circle1** a **circle2**. Do metody zkopírujte následující kód

      ```
      let dist = circle1.center.difference(circle2.center);
      let radius = circle1.radius + circle2.radius;
      return dist.squareSize < radius * radius;
      ```

   d. Vytvořte statickou metodu s názvem **checkRectCircleCollision**, která bude mít dva parametry **rect** a **circle**. Do metody zkopírujte následující kód

      ```
      let center = circle.center;
      let closePoint = new Vector(center.x, center.y);
      if (closePoint.x < rect.leftEdge) closePoint.x = rect.leftEdge;
      else if (closePoint.x > rect.rightEdge) closePoint.x = rect.rightEdge;
      if (closePoint.y < rect.topEdge) closePoint.y = rect.topEdge;
      else if (closePoint.y > rect.bottomEdge) closePoint.y = rect.bottomEdge;
      closePoint.subtract(center);
      return closePoint.squareSize < circle.radius * circle.radius;
      ```

8. Nyní vytvoříte třídu pro reprezentaci obdélníkových colliderů.
    a. Vytvořte třídu s názvem **RectCollider**, která bude dědit ze třídy **Positionable**.

    b. Uvnitř třídy vytvořte konstruktor, který bude mít tři parametry s názvy **position**, **width**, **height**.

       Parametr **position** předejte pomocí klíčového slova **super** konstruktoru rodičovské třídy. Parametry **width** a **height** uložte do atributů **this.width** a **this.height**.

    c. Vytvořte metodu s názvem **collideWith** s jedním parametrem **other**. Metoda vrátí zpět hodnotu **other.collideWithRect(this)**.

    d. Vytvořte metodu s názvem **collideWithRect** s jedním parametrem **other**. Metoda vrátí zpět hodnotu **CollisionDetection.checkRectCollision(this, other)**.

    e. Vytvořte metodu s názvem **collideWithCircle** s jedním parametrem **other**. Metoda vrátí zpět hodnotu **CollisionDetection.checkRectCircleCollision(this, other)**.

    f. Uvnitř třídy **RectCollider** vytvořte gettery s názvy **leftEdge**, **rightEdge**, **topEdge**, **bottomEdge**.

       Getter **leftEdge** bude vracet hodnotu **this.x**. Getter **rightEdge** bude vracet hodnotu **this.x + this.width**. Getter **topEdge** bude vracet hodnotu **this.y**. Getter **bottomEdge** bude vracet hodnotu **this.y + this.height**.

    g. Nyní bude třeba vytvořit settry se stejnými názvy, jako měly gettry z předchozího kroku. Všechny settry budou mít parametr **value**.

    h. Setter **leftEdge** přiřadí atributu **this.x** hodnotu **value**. Setter **rightEdge** přiřadí atributu **this.x** hodnotu **value - this.width**. Setter **topEdge** přiřadí atributu **this.y** hodnotu **value**. Setter **bottomEdge** přiřadí atributu **this.y** hodnotu **value - this.height**.

9. Nyní vytvoříte třídu pro reprezentaci kruhových colliderů.
    a. Vytvořte třídu s názvem **CircleCollider**, která bude dědit ze třídy **Positionable**.

    b. Uvnitř třídy vytvořte konstruktor, který bude mít dva parametry s názvy **position**, **radius**.

       Parametr **position** předejte pomocí klíčového slova super konstruktoru rodičovské třídy. Parametry **radius** uložte do atributu **this.radius**.

    c. Vytvořte metodu s názvem **collideWith** s jedním parametrem **other**. Metoda vrátí zpět hodnotu **other.collideWithCircle(this)**.

    d. Vytvořte metodu s názvem **collideWithRect** s jedním parametrem **other**. Metoda vrátí zpět hodnotu **CollisionDetection.checkRectCircleCollision(other, this)**.

e. Vytvořte metodu s názvem **collideWithCircle** s jedním parametrem **other**. Metoda vrátí zpět hodnotu **CollisionDetection.checkCircleCollision(other, this)**.

f. Uvnitř třídy **CircleCollider** vytvořte gettery s názvy **leftEdge**, **rightEdge**, **topEdge**, **bottomEdge**.

   Getter **leftEdge** bude vracet hodnotu **this.x**. Getter **rightEdge** bude vracet hodnotu **this.x + this.radius * 2**. Getter **topEdge** bude vracet hodnotu **this.y**. Getter **bottomEdge** bude vracet hodnotu **this.y + this.radius * 2**.

g. Nyní bude třeba vytvořit settry se stejnými názvy, jako měly gettry z předchozího kroku. Všechny settry budou mít parametr **value**.

   Setter **leftEdge** přiřadí atributu **this.x** hodnotu **value**. Setter **rightEdge** přiřadí atributu **this.x** hodnotu **value - this.radius * 2**. Setter **topEdge** přiřadí atributu **this.y** hodnotu **value**. Setter **bottomEdge** přiřadí atributu **this.y** hodnotu **value - this.radius * 2**.

h. Vytvořte getter s názvem **center**. Tento getter vrátí následující hodnotu **new Vector(this.x + this.radius, this.y + this.radius)**

10. Vyzkoušejte, zda třídy pro vyhodnocování kolizí fungují.
    a. Do souboru **index.js** vložte následující kód

```
let circle1 = new CircleCollider(new Vector(0, 0), 50);
let circle2 = new CircleCollider(new Vector(50, 50), 70);
let rect1 = new RectCollider(new Vector(110, 110), 50, 100);
let rect2 = new RectCollider(new Vector(150, 150), 100, 50);
console.log(circle1.collideWith(circle2));
console.log(rect1.collideWith(rect2));
console.log(circle2.collideWith(rect1));
console.log(rect1.collideWith(circle1));
```

    b. Program spusťte. V konzoli by se mělo objevit
       true
       true
       true
       false

11. Jako další budete vytvářet třídu Sprite, která bude sloužit pro snazší práci s obrázky, které budou patřit jednotlivým herním objektům.
    a. Vytvořte třídu s názvem **Sprite**, která bude rozšiřovat třídu **Positionable**.

    b. Ve třídě vytvořte konstruktor, který bude obsahovat čtyři parametry **imageId**, **position**, **width**, **height**.

       Hodnotu parametru **position** předejte jako argument konstruktoru rodičovské třídy.

       Hodnoty parametrů **width** a **height** uložte do atributů **this.width** a **this.height**.

Vytvořte atribut **this.image**, do kterého uložíte **document.getElementById(imageId)**

c. Ve třídě vytvořte metodu **draw** s parametrem **context**. Uvnitř metody **draw** zavolejte přes parametr **context** metodu **drawImage**. Argumenty metody **drawImage** budou **this.image**, **this.x**, **this.y**, **this.width**, **this.height**.

12. Dále budete vytvářet třídu, ze které budou dědit všechny herní objekty.
    a. Vytvořte třídu s názvem **GameObject**, která bude rozšiřovat třídu **Positionable**.

    b. Uvnitř třídy vytvořte konstruktor s jedním parametrem **position**, který předáte konstruktoru rodičovské třídy. Dále uvnitř konstruktoru vytvořte atribut **this.speed**, kterému přiřadíte hodnotu **new Vector(0, 0)**. Poté vytvořte atributy **this.minBounds** a **this.maxBounds**, kterým přiřadíte hodnoty **new Vector(-Infinity, -Infinity)** a **new Vector(Infinity, Infinity)**.

    c. Uvnitř třídy vytvořte metodu **move**. Uvnitř metody **move** zavolejte metodu **add** atributu **this.position** a jako argument metody **add** použijte atribut **this.speed**.

    d. Poté vytvořte metodu **draw** s parametrem **context**. Do metody vložte následující příkaz: **this.sprite.draw(context)**.

    e. Následně vytvořte metodu **collideWith** s parametrem **other**. Do metody vložte následující příkaz: **return this.collider.collideWith(other.collider)**.

13. Po vytvoření třídy **GameObject** vytvořte třídu **Player**.
    a. Vytvořte třídu **Player**, která bude dědit ze třídy **GameObject**.

    b. Vytvořte konstruktor třídy s parametry **position**, **size**.

    Hodnotu parametru **position** předejte konstruktoru rodičovské třídy.

    Vytvořte atribut **this.standingSprite**, do kterého uložíte novou instanci třídy **Sprite**. Argumenty konstruktoru nové instance budou **"player_standing"**, **position**, **size**, **size**.

    Dále vytvořte atribut **this.crouchingSprite**, do kterého uložíte taktéž novou instanci třídy **Sprite**. Argumenty při vytváření této instance budou **"player_crouching"**, **position**, **size * 1.1**, **size * 0.75**.

    Následně vytvořte atribut **this.sprite** a uložte do něj hodnotu **this.standingSprite**.

    Poté vytvořte atribut **this.standingCollider** do nějž uložíte novou instanci třídy **CircleCollider**. Argumenty nové instance třídy **CircleCollider** budou **position**, **size / 2**.

    Poté vytvořte atribut **this.crouchingCollider** do nějž uložíte novou instanci třídy **RectCollider** s argumenty **position**, **size * 1.1**, **size * 0.75**.

    Poté vytvořte atribut **this.collider** a uložte do něj hodnotu **this.standingCollider**.

Nakonec uvnitř konstruktoru vytvořte atributy **this.crouching** a **this.grounded** do kterých uložte hodnotu **false**.

c. Po dokončení konstruktoru třídy vytvořte metodu **jump**.

   Uvnitř metody vytvořte podmínku, která bude kontrolovat, zda je hodnota atributu **this.grounded** rovna **true**.

   Pokud ano, nastavte hodnotu atributu **this.speed.y** na **-14**.

d. Poté vytvořte metodu s názvem **getUp**.

   Uvnitř metody vytvořte podmínku, která bude kontrolovat, zda je hodnota atributu **this.crouching true**.

   Pokud ano nastavíte hodnotu atributu **this.collider** na **this.standingCollider**, **this.sprite** na **this.standingSprite** a **this.crouching** na **false**.

e. Dále vytvořte metodu **crouch**.

   Uvnitř metody vytvořte podmínku, která bude kontrolovat, zda je hodnota atributu **this.crouching true**.

   Pokud ano navrátíte se pomocí slova return z funkce.
   Pod podmínkou ve funkci **crouch** nastavte hodnotu atributu **this.crouching** na **true**, **this.collider** na **this.crouchingCollider**, **this.sprite** na **this.crouchingSprite** a k atributu **this.y** přičtěte hodnotu **this.standingCollider.bottomEdge - this.crouchingCollider.bottomEdge**.

f. Nakonec ve třídě **Player** vytvořte metodu **move**.

   Uvnitř metody zvyšte hodnotu atributu **this.speed.y** o **0.3**.

   Poté pomocí slova super zavolejte metodu **move** rodičovské třídy.

   Po zavolání metody z rodičovské třídy vytvořte podmínku, která bude kontrolovat pravdivost následujícího výrazu: **this.collider.bottomEdge > this.maxBounds.y**.

   Pokud bude podmínka splněna nastavíte hodnotu atributu **this.collider.bottomEdge** na **this.maxBounds.y**, **this.speed.y** na **0** a **this.grounded** na **true**.

   Pokud podmínka splněna nebude nastavíte **this.grounded** na **false**.

14. Vyzkoušejte, že nově vytvořené třídy fungují.
    a. Uvnitř souboru index.js vytvořte proměnnou **player** do které uložte novou instanci třídy **Player** (argumenty mohou být například takovéto **new Vector(100, 0)**, **150**).

b. Proměnnou **player** přidejte jako další argument metody **draw** proměnné **scene** (**scene.draw(background, player)**)

c. Otevřete aplikaci v prohlížeči. Na plátně by měl být vykreslen kromě pozadí i obrázek dinosaura.

15. Jako další vytvoříte třídu, která bude sloužit pro vytváření kruhových překážek s obrázkem meteoritu.
    a. Vytvořte třídu s názvem **Meteor**, která bude rozšiřovat třídu **GameObject**.

    b. Vytvořte konstruktor třídy s parametry **position**, **speed**, **size**.

       Parametr **position** předejte do konstruktoru rodičovské třídy.

       Parametr speed uložte do atributu **this.speed**.

       Vytvořte atribut **this.sprite** a uložte do něj novou instanci třídy **Sprite**. Argumenty konstruktoru budou **"meteor"**, **position**, **size**, **size**.

       Vytvořte atribut **this.collider** a uložte do něj novou instanci třídy **CircleCollider** s argumenty **position**, **size / 2**.

    c. Uvnitř třídy Meteor vytvořte metodu **move**. Uvnitř metody zavolejte metodu **move** ze třídy rodičovské (použijete k tomu klíčové slovo **super**).

       Poté v metodě vytvořte podmínku kontrolující pravdivost následujícího výrazu **this.collider.rightEdge < this.minBounds.x**.

       Při splnění této podmínky zavolejte metodu **this.onSceneExit**.

    d. Vyzkoušejte funkčnost vytvořené třídy (Zkoušení funkčnosti bude probíhat stejně jako u třídy **Player**)

16. Dále vytvoříte třídu, která bude sloužit pro vytváření obdélníkových překážek s obrázkem kaktusu.
    a. Vytvořte třídu s názvem **Cactus**, která bude rozšiřovat třídu **GameObject**.

    b. Vytvořte konstruktor třídy s parametry **position**, **speed**, **width**, **height**. Parametr **position** předejte do konstruktoru rodičovské třídy.

       Parametr speed uložte do atributu **this.speed**.

       Vytvořte atribut **this.sprite** a uložte do něj novou instanci třídy **Sprite**. Argumenty konstruktoru budou **"cactus"**, **position**, **width**, **height**.

       Vytvořte atribut **this.collider** a uložte do něj novou instanci třídy **RectCollider** s argumenty **position**, **widht**, **height**.

c. Ze třídy Meteor zkopírujte metodu **move** a vložte ji do třídy **Cactus**.

d. Vyzkoušejte funkčnost vytvořené třídy.

17. Poslední vytvářenou třídou bude třída pro vykreslování skóre.
    a. Vytvořte třídu s názvem **Score**, která bude rozšiřovat třídu **Positionable**.

    b. Vytvořte konstruktor třídy s parametry **position**, **font**, **size**.

       Parametr **position** předejte do konstruktoru rodičovské třídy.

       Vytvořte atribut **this.value** a uložte do něj hodnotu **0**.

       Vytvořte atribut **this.font** a uložte do něj hodnotu následujícího výrazu **size + "px" + " " + font**.

    c. Uvnitř třídy vytvořte metodu **increase**. V této metodě zvyšte hodnotu atributu **this.value** o **1**.

    d. Dále vytvořte metodu **draw** s parametrem **context**.

       V této metodě nejprve nastavte atribut font parametru **context** na hodnotu **this.font**.

       Poté přes parametr **context** zavolejte metodu **fillText** a jako argumenty použijte **this.value**, **this.x**, **this.y**.

18. V poslední fázi bude potřeba vytvořené třídy pospojovat a vytvořit fungující aplikaci.
    a. Nejprve smažte vše, co máte v souboru **index.js**.

    b. Vytvořte v souboru index.js proměnnou **scene** a uložte do ní novou instanci třídy **Scene**. Argumenty konstruktoru dejte **1100**, **700**.

    c. Vytvořte proměnnou **background** a uložte do ní novou instanci třídy **Background**. Argumenty konstruktoru budou: **scene.width**, **scene.height**.

    d. Vytvořte proměnnou **score** a uložte do ní novou instanci třídy **Score**. Argumenty konstruktoru budou: **new Vector(scene.width / 2, 50), "Arial", 30**.

    e. Vytvořte proměnnou **player** a uložte do ní novou instanci třídy **Player**. Argumenty konstruktoru budou: **new Vector(100, 0), 150**.

    f. Po vytvoření instance třídy **Player** nastavte vytvořené instanci atribut **maxBounds** na **new Vector(scene.width, scene.height * 0.87)**.

    g. Vytvořte funkci s názvem **randomObstacle**, která bude mít jeden parametr s názvem **xPos**.

Uvnitř funkce vytvořte podmínku, která zkontroluje, zda je hodnota výstupu z metody **Math.random** větší než **0.5**.

Pokud ano vytvořte novou proměnnou s názvem **obstacle**, která bude globální pro kontext funkce **randomObstacle** a uložíte do ní novou instanci třídy **Meteor** s argumenty: **new Vector(xPos, scene.height * 0.5), new Vector(-5, 0), 140**.

Pokud podmínka splněna nebude vytvoříte proměnnou se stejným názvem a rozsahem jako když podmínka splněna byla, akorát do ní uložte novou instanci třídy **Cactus** s argumenty: **new Vector(xPos, scene.height * 0.7), new Vector(-5, 0), 100, 100**.

Ve funkci **randomObstacle**, pod bloky podmínek nastavte proměnné **obstacle** atribut **minBounds** na **new Vector(0, 0)** a atribut **maxBounds** na **new Vector(scene.width, scene.height)**.

Nakonec proměnnou obstacle z funkce vraťte.

h. Po vytvoření funkce **randomObstacle** vytvořte proměnnou **obstacle1** a uložte do ní výstup z funkce **randomObstacle**. Jako atribut při volání této funkce použijte hodnotu **scene.width**.

i. Stejným způsobem jako u proměnné **obstacle1** vytvořte proměnnou **obstacle2** akorát jako argument funkce **randomObstacle** použijte hodnotu **scene.width * 1.5**.

j. Vytvořte funkci s názvem **onObstacle1Exit**, která nebude mít žádný parametr. Ve funkci nejprve přiřadíte proměnné **obstacle1** hodnotu **randomObstacle(scene.width)** a poté nastavíte atribut **obstacle1.onSceneExit** na hodnotu **onObstacle1Exit** (do atributu uložíte samotnou funkci ne výstup z jejího volání).

k. Vytvořte funkci s názvem **onObstacle2Exit**, která nebude mít žádný parametr. Ve funkci nejprve přiřadíte proměnné **obstacle2** hodnotu **randomObstacle(scene.width)** a poté nastavíte atribut **obstacle2.onSceneExit** na hodnotu **onObstacle2Exit** (do atributu uložíte samotnou funkci ne výstup z jejího volání).

l. Po vytvoření funkcí **onObstacle1Exit** a **onObstacle2Exit** přiřaďte tyto funkce jako hodnoty atributů **obstacle1.onSceneExit** a **obstacle2.onSceneExit** (stejně jako to děláte ve funkcích samotných)

m. Vytvořte proměnnou s názvem **keys** a uložte do ní prázdné pole (proměnná se bude rovnat hodnotě **[]**). Pomocí této proměnné budete zjišťovat, jaké klávesy jsou stisknuty.

n. Do atributu **document.onkeydown** uložte novou funkci s jedním parametrem, který bude mít název **event** (funkci můžete vytvořit jako normální funkci s názvem například onKeyDown, ale úplně postačí, když funkci vytvoříte jako anonymní).

Uvnitř funkce, kterou jste uložili do atributu **onkeydown** objektu **document** vložte následující kód: **keys[event.key] = true;**

o. Do atributu **document.onkeyup** uložte novou funkci s jedním parametrem, který bude mít název event (funkci můžete vytvořit jako normální funkci s názvem například **onKeyUp**, ale úplně postačí, když funkci vytvoříte jako anonymní).

Uvnitř funkce, kterou jste uložili do atributu **onkeyup** objektu **document** vložte následující kód: **keys[event.key] = false;**

p. Vytvořte funkci s názvem **gameLoop**, která nebude mít žádný parametr.

Ve funkci zavolejte přes proměnnou **score** metodu **increase**.

Poté do funkce vložte následující podmínky:
if (keys["s"]) player.crouch();
else player.getUp();
if (keys["w"]) player.jump();

Poté zavolejte metody **move** proměnných **player**, **obstacle1** a **obstacle2**.

Následně, pomocí metody **draw**, **ktrerou** zavoláte přes proměnnou **scene**, vykreslete scénu. Argumenty této metody budou: **background**, **player**, **obstacle1**, **obstacle2**, **score**.

Poté vytvořte podmínku, která bude kontrolovat, zda nedochází ke kolizím mezi hráčem a překážkami (K vyhodnocení kolizí použijte metodu **collideWith**, která je všech instancích třídy **GameObject**.). Pokud bude ke kolizi docházet opusťte funkci **gameLoop** pomocí příkazu **return**.

Nakonec do funkce **gameLoop** vložte následující příkaz **requestAnimationFrame(gameLoop)**.

Po dokončení funkce **gameLoop** funkci zavolejte (Hlavně ji zavolejte mimo všechny funkce, ne že ji zavoláte ve funkci samotné.)

19. Poslední úkol už bude bez popisu jednotlivých kroků. Vaším úkolem bude využít obrázky, kde je pozadí rozloženo do více vrstev a předělat třídu **Background** takovým způsobem, aby vykreslovala pozadí s efektem parallax.