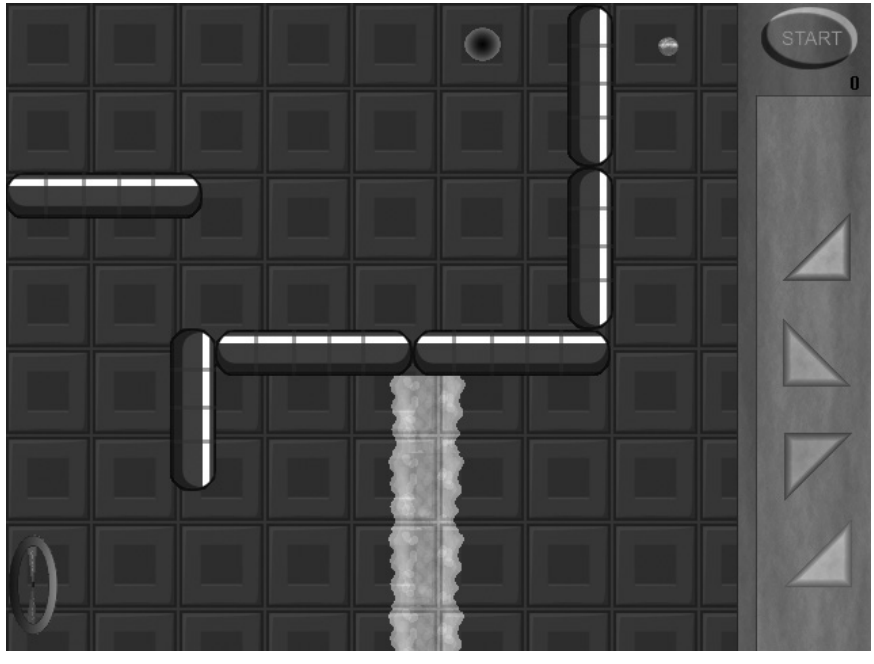
Kapitola 4

Vytváříme logickou hru

Jak již z jejich označení vyplývá, jsou tyto hry určené pro hráče s oblibou v rébusech, hlavolamech a hádankách. K jejich úspěšnému dohrání tak nestačí jen štěstí či rychlé reflexy. Tyto hry, ačkoli se to může zdát nepravděpodobné, často udrží hráče na mnohem delší dobu než na pohled zábavnější žánry.

Dnes jsou logické hry snad nejméně zastoupeným žánrem – hráči o ně zkrátka již nemají zájem. Možná právě proto, že na rozdíl od akčních her vyžadují často dlouhé přemýšlení a dostatek důvtipu.

My společně vytvoříme jednodušší hru, ovšem ukážeme si využití např. goniometrických funkcí či ovlivňování pohybu prací se souřadnicemi a tak půjde – z hlediska vývoje – o skutečně složitější hru než byly ty předešlé.



Obrázek 4.1: Naše hra po spuštění bez rozmístěných překážek

Principem hry bude dostat kuličku z počátečního bodu do jamky, čemuž budou bránit různé překážky. K jejich překonání bude hráč muset umístit omezený počet různě tvarovaných bariér a kuličku tak pomocí odrazů dopravil do cíle. Míček bude moci být unášen vodou, ovlivněn větrem či vyskočit na skokánku.

Hlavní rozvržení a základní nastavení

Protože jsme v předchozích kapitolách již dostatečně vyzkoušeli tvorbu všech ostatních částí hry, nyní se více zaměříme pouze na hlavní nabídku a hru samotnou.

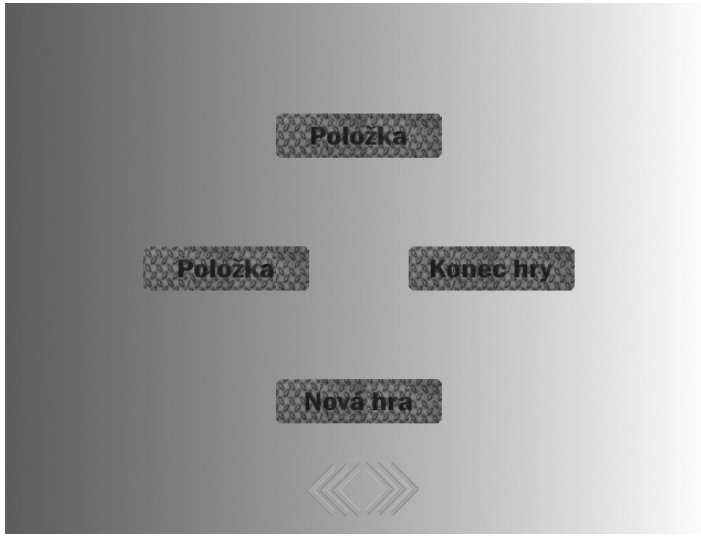
Hra poběží v okně s rozměry 640 x 480, bez zobrazení předdefinované nabídky a bude tvořena dvěma snímky – jeden pro hlavní nabídku a druhý pro obrazovku hry.

Po vytvoření nové aplikace si tedy zobrazíme její **Properties**. Pod ikonou **Window** () vyškrtneme položku **Menu bar**.

Ve **Storyboard Editoru** tedy vytvoříme druhý snímek a můžeme se přepnout do první obrazovky pro její úpravy.

Hlavní nabídka

Zatím jsme pro volbu jednotlivých položek používali vždy myš nebo klávesnici, teď tomu bude jinak. Nyní budou jednotlivé položky umístěny v kruhu, kterým bude uživatel otáčet klepnutím na tlačítko směru. Pohyb a umístění v kruhu však budou kompletně počítány pomocí matematických funkcí.



Obrázek 4.2:
Nabídka s otáčejícími se položkami je poměrně obtížná

Nejprve vytvoříme pozadí, které bude tvořeno barevným přechodem. Tuto funkci přímo nabízí objekt **Quick Backdrop**.

1. Vložíme jej tedy na plochu snímku a zobrazíme jeho **Properties**.
2. Pod ikonou **Settings** () klepneme na řádek **Type** a změníme volbu na **Gradient**.
3. Zaškrtneme volbu **Vertical Gradient**, protože chceme přechod barvy odshora dolů.
4. Podle libosti ponecháme či změníme hodnotu barev poklepáním a výběrem.
5. Klepneme na ikonu **Size/Position** ().
6. Hodnoty **X** a**Y** nastavíme na 0, hodnoty řádku **Width** a **Height** pak na 640 a 480.

Teď vložíme jednotlivé položky a tlačítka, jimiž se bude ovládat posun (resp. otáčení) celé nabídky.

1. Vložíme 4 aktivní objekty a z adresáře Zdroje\Logicka použijeme grafiku ze souborů logNab1.PNG, logNab2.PNG, logNab3.PNG a logNab4.PNG a **Hot Spot** umístíme vždy do středu.
2. Poté vytvoříme další dva aktivní objekty, pro něž použijeme jako zdroj grafiky soubory Zdroje\Logicka\logPrava.PNG a Zdroje\Logicka\logLeva.PNG.
3. Oběma těmto objektům v jejich **Properties** pod položkou **Runtime Options** () vypneme možnost **Use fine detection**.

Položky nabídky nemusíme nijak řadit či rozmisťovat, protože jejich pozici i pohyb bude kontrolovat program. Tlačítka pro otáčení umístíme do středu obrazovky dolů jako na obrázku výše.

Přepneme se tedy do **Event editoru** pro vytvoření potřebných událostí. První z nich slouží k vypočítání souřadnic položek pro uspořádání do kruhu.

1. Vložíme novou podmínku z objektu **Special** () a vybereme možnost **Always**.
2. U první položky zvolíme **Position→Set X Coordinate** a vepíšeme `Cos(Global Value A)*120+320`.
3. Druhou akcí bude **Position→Set Y Coordinate**, přičemž nyní zadáme výraz `Sin(Global Value A)*120+240`.
4. Následující položce nabídky přiřadíme stejné akce, ovšem k hodnotě A budeme přičítat číslo 90. Výraz pro nastavení X pozice bude mít tedy tvar `Cos(Global Value A+90)*120+320` a druhý, určující souřadnici Y, pak `Sin(Global Value A+90)*120+240`.
5. U třetího objektu pak budeme přičítat číslo 180 a u čtvrtého 270.

Každý objekt má tedy přiřazeny dvě akce pro nastavení souřadnic X a Y, jejichž zápis se liší jen velikostí čísla přičítaného k hodnotě globální proměnné A. Pokud nyní snímek spustíme, uvidíme položky umístěné v kruhu. Jak ale vlastně ona událost funguje?

Předně musím uvést, že v rámci této knihy se nemohu zabývat významem a vysvětlením goniometrických funkcí sinus a cosinus, které jsou ve výrazech výše obsaženy. Z toho důvodu mohou čtenáři bez alespoň základní představy o této problematice následující tři odstavce s klidným svědomím vynechat.

Globální hodnota A je zpočátku rovna nule – funkce ve výrazech jednotlivých položek tedy počítají s hodnotami 0, 90, 180 a 270. Protože kruh má 360°, čtyři položky do něho mohou být pravidelně rozmístěny jen v případě, že funkce na výpočet souřadnic použijí pro každou z nich hodnotu o 90° vyšší (či nižší) než měla předchozí.

Číslo 120, kterým je výsledek funkce v každém výrazu násoben, vlastně společně udávají vzdálenost od středu – čím vyšší toto číslo bude, tím větší kruh položky vytvoří. Technicky vzato, sinus a cosinus nabývají hodnot pouze od -1 do 1 a tak pokud jsem uvedl, že to v podstatě udává velikost výsledného kruhu, je násobení snadno pochopitelné.

Každý výraz pro souřadnici X končí přičítáním čísla 320 a pro souřadnici Y pak 240. To proto, aby byly objekty posunuté doprostřed snímku (640/2=320 a 480/2=240). Bez toho by byly výsledky záporné a objekty by se dostávaly mimo obrazovku.

Teď se můžeme pustit do kruhového pohybu položek. To bude zajištěno jednoduše – přičítáním nebo odečítáním ke globální hodnotě A, čímž se bude plynule měnit výsledek funkcí. Vzdálenost mezi objekty se však měnit nebude, protože hodnoty 90, 180 a 270 jsou přičítány nezávisle a rozdíl tak bude stále stejný.

1. Vytvoříme podmínku z objektu **The mouse pointer and keyboard** () a zvolíme **The mouse→User clicks on an object**.
2. Zvolíme objekt šipky doleva a potvrdíme.
3. Akci přiřadíme objektu **Special** (), kde vybereme **Change a global value→ Set**.
4. Vybereme **Global Value B** a zadáme číslo 1.
5. Vytvoříme další podmínku a z objektu **Special** () zvolíme **Compare to a global value**.
6. Zvolíme **Global Value B** a zadáme hodnotu 1.
7. Akce nastavíme pod objektem **Special** () – **Change a global value→Add to**.
8. Necháme zvolenou proměnnou A a vepíšeme číslo 1.
9. Vytvoříme druhou – stejnou – akci a zvolíme proměnnou C.

Je-li hodnota globální proměnné B=1, pak se bude k proměnným A a C plynule přidávat číslo 1. Při výpočtech pozice se hodnoty po sobě jdoucích položek liší o 90, tedy k posunu na další z nich je nutné toto číslo přičíst či odečíst. Proto se současně přidává i hodnota C, kterou při dosažení hodnoty 90 vynulujeme spolu s hodnotou B, aby se otáčení zastavilo.

1. Vytvoříme podmínku a z objektu **Special** () zvolíme **Compare to a global value**.
2. Zvolíme **Global Value C** a vepíšeme hodnotu 90.
3. Nyní nastavíme pod objektem **Special** () akci **Change a global value→Set**.
4. Zvolíme **Global Value B** a potvrdíme s nulovou hodnotu.
5. Opakujeme kroky 3 a 4, zvolíme však **Global Value C**.

Nyní vytvoříme otáčení na opačnou stranu – jediný rozdíl bude v tom, že se od hodnoty A bude číslo 1 odečítat, takže výsledek se bude snižovat.

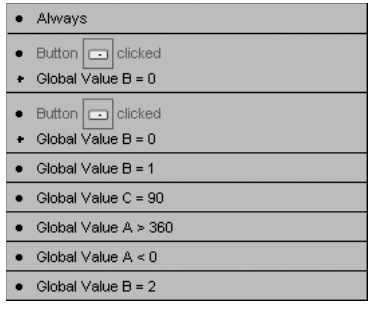
1. Podmínku vytvoříme z objektu **The mouse pointer and keyboard** () a zvolíme **The mouse→User clicks on an object**
2. Vybereme objekt šipky doprava.
3. Pod objektem **Special** () zvolíme **Change a global value→Set**.
4. Vybereme **Global Value B** a zadáme hodnotu 2 (1 tedy určuje pohyb doleva, 2 doprava).
5. Další podmínku vytvoříme z objektu **Special** () a vybereme **Compare to a global value**.
6. Zvolíme **Global Value B**, vepíšeme číslo 2 a potvrdíme.
7. Pod objektem **Special** () nastavíme **Change a global value→Subtract from**.
8. Ponecháme nastaveno **Global Value A** a vepíšeme číslo 1.
9. Pod tímtéž objektem ještě nastavíme **Change a global value→Add to**, zvolíme **Global Value C** a zadáme číslo 1.

Aby program věděl, která možnost je vybrána, bude se dívat na hodnotu globální proměnné A. Ta má pro jednotlivé položky hodnoty 0, 90, 180 nebo 270. Pokud však hráč nechá nabídku otočit víckrát, hodnota překročí 360 či klesne pod 0 a výběr přestane být funkční. To jednoduše ošetříme:

1. Podmínku vytvoříme z objektu **Special** () a vybereme **Compare to a global value**.
2. Ponecháme vybranou hodnotu A.
3. Operátor změníme na **Greater** a vepíšeme číslo 359.
4. Pod stejným objektem zvolíme akci **Change a global value→Set** a potvrdíme nulovou hodnotu proměnné A.
5. Vytvoříme další podmínku a opakujeme kroky 1 a 2.
6. Nyní operátor změníme na **Lower**.
7. Pomocí **Change a global value→Set** nastavíme hodnotu **Global Value A** na 359.

Poslední úpravou samotného pohybu bude znemožnění použití tlačítek, pokud se nabídka hýbe. Víme, že pohyb nastává změnou hodnoty A, která pak nastává jen pokud je B=1 nebo 2 (tedy pokud se A snižuje či zvyšuje).

1. Do podmínky klepnutí na objekt šipky vpravo vložíme vnořenou podmínku z objektu **Special** ( ) – **Compare to a global value**.
2. Zvolíme **Global Value B** a potvrdíme rovnost nulové hodnotě.
3. Stejně omezíme i podmínku klepnutí na objekt šipky vlevo.



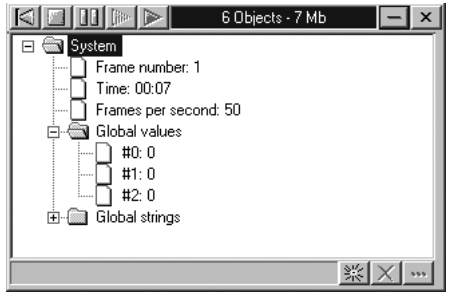
Obrázek 4.3: **Všechny nutné podmínky pro pohyb hlavní nabídky**

Zbývá vytvořit potvrzení vybrané volby. Protože máme více položek než budeme potřebovat, stačí vytvořit jen dvě podmínky – pro ukončení a pro přechod do hry. V závislosti na tom, jak máme položky seřazené, budou se lišit i hodnoty A pro nastavení daných akcí. Musíme tedy zjistit, jakou hodnotu má která položka.

K tomu nám poslouží vestavěný *Debugger*, což je pomůcka umožňující zjištění prakticky všech údajů v běžící aplikaci.

Obrázek 4.4: **Debugger aplikace**

1. Spustíme hru a klepnutím na tlačítko **Show or Hide** ( ) v **Debuggeru** jej otevřeme. Klepnutím otevřeme položku **System** a poté **Global value**.



Obrázek 4.5: **Zobrazení globálních hodnot v Debuggeru**

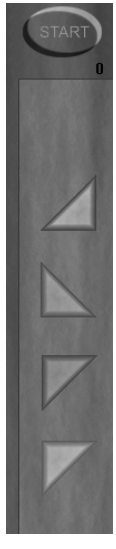
2. Globální hodnoty jsou zde zobrazeny jako čísla, tedy #0 představuje globální proměnnou A, jejíž hodnotu sledujeme.
3. Otočíme nabídkou tak, aby byla zvolena položka Konec hry.
4. Zapamatujeme si hodnotu zobrazenou v **Debuggeru,** otočíme na položku Nová hra a opět zjistíme aktuální hodnotu globální proměnné A (#0).
5. Ukončíme hru a přepneme do Event editoru.
6. Vložíme novou podmínku z objektu **Special** ( ) a zvolíme **Compare to a global value**.
7. Zadáme první hodnotu, kterou jsme viděli při zvolení položky Konec hry a akci nastavíme pod objektem **Storyboard controls** ( ), kde zvolíme **End application**.
8. Opakujeme kroky 6–7 a zadáme hodnotu pro Novou hru.
9. Nyní objektu **Storyboard controls** ( ) zvolíme **Next frame**.

Obrazovku nabídky máme kompletní, můžeme se tedy přepnout do úprav druhého snímku.

Rozhraní hry

Obrazovka bude tvořena herní plochou a postranním panelem. Ten bude kromě informací obsahovat hlavně objekty odrazových ploch, jejichž vhodné umístění hráčem je cílem hry.

1. Vložíme objekt **Backdrop** a použijeme grafiku ze souboru Zdroje\Logicka\logPanel.PNG.
2. Objektu v jeho **Properties** nastavíme pozici X na hodnotu 540, Y na 0.

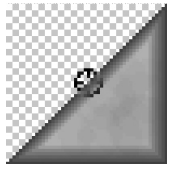


Obrázek 4.6
Umístěný herní panel s dalšími prvky

3. Nyní vložíme další object **Backdrop**, grafiku nahrajeme ze souboru Zdroje\Logicka\logLista.PNG.
4. V Properties pod ikonou **Runtime Options** ( ) klepneme na řádek **Obstacle Type** a zvolíme **Obstacle**.
5. Poté i tomuto objektu nastavíme pozici X na 540 a Y na 0.

Máme vytvořen panel, na němž se budou nacházet objekty, tlačítko pro start hry a počitadlo, zaznamenávající počet kroků, tedy kolikrát hráč celkově manipuloval s odrazovými plochami. Ty si teď vytvoříme:

1. Vložíme aktivní objekt a jeho grafiku načteme ze souboru Zdroje\Logicka\logP1.PNG.
2. **Hot Spot** umístíme vždy do středu zkosené strany mírně od okraje jako na obrázku níže.



Obrázek 4.7:
Vhodné umístění Hot Spotu

3. V Properties objektu pod ikonou **Events** ( ) klepneme na řádek **Qualifier(s)**, kde zvolíme **Edit**.
4. Klepneme na **Add** a vybereme libovolný symbol, jenž musí být ovšem stejný i pro ostatní objekty.
5. Po potvrzení objekt umístíme podobně jako na obrázku tak, aby se nedotýkal lišty panelu.
6. Vytvoříme další 3 aktivní objekty, grafiku použijeme ze souborů logP2.PNG, logP3.PNG a logP4.PNG z adresáře Zdroje\Logicka a u každého uplatníme kroky 2–5.

Dále vložíme aktivní objekt pro objekt startu a použijeme grafiku ze souboru logStart.PNG. Poté vložíme poslední objekt – počitadlo. Oba objekty poté umístíme podobně jako na obrázku 4.1.

Manipulace a umisťování objektů

Hráč bude objekty přemisťovat tak, že je klepnutím myši „sebere“ a poté dalším klepnutím nechá na zvoleném místě. Počet možných přesunů každého objektu necháme neomezený, přičemž hráč bude mít možnost vrátit objekt zpět na plochu panelu.

Sbírání a pokládání bude fungovat tak, že po klepnutí na objekt se jeho vnitřní přepínač nastaví na opačnou hodnotu. Jedna z hodnot pak bude v samostatné události přikazovat objektu, aby svou pozici neustále nastavoval na pozici kurzoru myši. Jakmile však hráč klepne se „sebraným“ objektem, přepínač se vrátí na původní hodnotu a objekt již nebude myš následovat – zůstane tedy na místě.

1. V **Event editoru** vytvoříme novou podmínku z objektu **The mouse pointer and keyboard** (), kde zvolíme **The mouse→User clicks on an object**.
2. V seznamu objektů nyní vybereme zvolený symbol zástupce skupiny.
3. Vložíme vnořenou podmínku a z objektu zástupce skupiny zvolíme **Collisions→Overlapping a backdrop**.
4. Tuto podmínku negujeme.
5. Nyní vložíme druhou vnořenou podmínku a opět pod objektem zástupce skupiny zvolíme **Collisions→Overlapping another object**.
6. V seznamu objektů zvolíme zástupce skupiny a potvrdíme.
7. Také tuto podmínku negujeme, takže výsledná celková podmínka bude vypadat jako na obrázku níže, samozřejmě podle zvoleného symbolu zástupce skupiny.
8. Akci přiřadíme zástupci skupiny, zvolíme **Flags→Toggle** a zadáme číslo 1.

Obrázek 4.8:
Podmínka pro „sebrání" či „položení" objektu

První vnořená podmínka říká, že objekt se nesmí dotýkat pozadí. To proto, aby jej hráč nemohl umístit zčásti na herní panel a zčásti na plochu hry – naše lišta na kraji panelu má z toho důvodu aktivovaný stav pozadí, resp. překážky, takže objekt se jí nesmí dotýkat. Pokud jsme ji umístili od počátku tak, že se lišty dotýká, nepůjde nám „sebrat". Tato podmínka se bude hodit i později, až do herního prostředí umístíme překážky, které budou nastaveny také jako pozadí.

Druhá vnořená podmínka vyjadřuje, že objekt se nesmí dotýkat jiného objektu ze skupiny. Tím jsme ošetřili možnost, kdy se hráč pokouší jeden objekt položit na druhý.

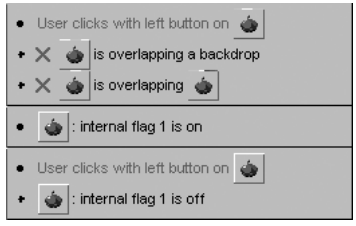
Teď je třeba vytvořit událost, při které bude objekt se zapnutým přepínačem stále umisťován na stejné souřadnice, jaké má kurzor myši.

1. Vytvoříme novou podmínku ze zástupce skupiny a vybereme **Alterable Values→Flags→Is a flag on?**.
2. Zadáme číslo 1 a potvrdíme.
3. Zástupci skupiny nejprve přiřadíme akci **Position→Set X Coordinate** a vepíšeme `Xmouse`.
4. Druhou akcí pak bude samozřejmě **Position→Set Y Coordinate**, kam vepíšeme `Ymouse`.

Poslední událost týkající se manipulace s objekty je počítání jednotlivých kroků.

1. Vytvoříme podmínku z objektu **The mouse pointer and keyboard** (), kde zvolíme **The mouse→User clicks on an object**.
2. Vybereme symbol zástupce skupiny a potvrdíme.
3. Teď vložíme vnořenou podmínku ze zástupce skupiny a zvolíme **Alterable Values→Flags→Is a flag off?**.
4. Vepíšeme číslo 1.

5. Objektu počitadla přiřadíme akci **Add to Counter** a vepíšeme číslo 1.



Obrázek 4.9:
Všechny podmínky pro manipulaci s objekty

Vybavení úrovně

Mezi vybavení úrovně patří nejen pozadí a překážky, ale také objekt kuličky, která je hlavním prvkem celé hry.

Nejprve vložíme pozadí, které bude stejně jako v případě plošinové hry tvořené vzorkováním pomocí objektu **Quick backrop**.

1. Vložíme tedy tento objekt a zobrazíme jeho **Properties**.
2. Pod ikonou **Settings** ( ) klepneme na řádek **Type** a vybereme volbu **Motif**.
3. Klepneme na tlačítko **Edit** a vložíme grafiku ze souboru Zdroje\Logicka\logQ.PNG.
4. Pod ikonou **Size/Position** ( ) nastavíme hodnoty **X** a **Y**, oboje na 0, **Width** a **Height** na 640 a 480, aby pozadí vyplnilo celou plochu snímku.

Jak vidíme, pozadí nám nyní překrylo všechno ostatní kromě aktivních objektů. Klepneme na něj tedy pravým tlačítkem myši a zvolíme **Order→To Back**.

Nyní do úrovně umístíme překážky:

1. Vytvoříme dva objekty **Backdrop** a použijeme grafiku ze souborů Zdroje\Logicka\logPrek1.PNG a Zdroje\Logicka\logPrek2.PNG.
2. Zobrazíme si **Properties** a klepneme na ikonu **Runtime Options** ( ).
3. V řádku **Obstacle Type** zvolíme u obou objektů možnost **Obstacle**.
4. Z obou objektů pomocí jejich kopírování vytvoříme pokud možno podobnou stavbu jako na obrázku 4.1.

Vložíme ještě objekt jamky, jehož jediný účel je poté, co se jej dotkne kulička, přejít na další snímek – zatím druhou úroveň nemáme a dojde tedy k ukončení aplikace. Vytvoříme aktivní objekt, použijeme grafiku ze souboru Zdroje\Logicka\logJamka.PNG a objekt umístíme podle obrázku 4.1.

Zbývá objekt kuličky:

1. Vložíme aktivní objekt a grafiku použijeme ze souboru Zdroje\Logicka\logKulicka.PNG.
2. **Hot Spot** umístíme na střed.
3. V **Properties** klepneme na ikonu **Movement** ( ) a v řádku **Type** zvolíme možnost **Bouncing ball**.

4. Klepneme na **Initial direction** a zvolíme jen směr dolů.
5. V řádku **Speed** zadáme hodnotu 40.
6. Vyškrtneme volbu **Moving at start**.
7. **Randomizer** nastavíme na hodnotu 0 a **Security** na 10.

Volba **Randomizer** udává, nakolik přesné mají být odrazy. Čím je hodnota tohoto nastavení vyšší, tím více se budou odrazy lišit. Při nízkém nastavení pak budou co možná nejskutečnější.

Nastavení **Security** zabraňuje vzniku situace, kdy by se objekt mezi dvěma překážkami začal odrážet donekonečna. Program rozpozná, že objekt se odráží stále stejně a po určité době jej nechá odrazit jinak a smyčku tak ukončí. Pro nás je tato možnost jen málo významná.

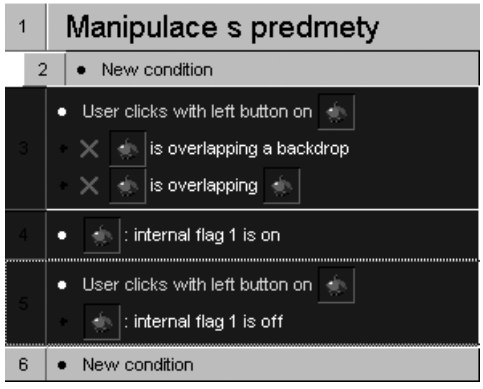
Pohyb kuličky

Klepne-li hráč na tlačítko start, kulička se dá do pohybu a již nebude možné s překážkami manipulovat. Při kontaktu s pozadím či objekty odrazových ploch dojde k jejímu odrazu, při doteku s jamkou.

Jako první vytvoříme aktivování pohybu kuličky a znemožnění dalšího přesunu překážek. K tomu využijeme dosud neznámou věc a to je uzavírání událostí do skupin a podskupin.

Skupiny jsou nesmírně užitečné a umožňují vlastně vypínat a opět zapínat celé části programu. Nemusíme tedy vytvářet složité podmínky, kdy je to či ono možné – jednoduše nežádoucí podmínky deaktivujeme. A to uděláme právě teď.

1. V **Event editoru** klepneme pravým tlačítkem myši na číslo první podmínky.
2. Zvolíme **Insert→A group of events**.
3. Zadáme název skupiny – například „Manipulace s objekty“ a potvrdíme. Řádky níže jsou pro zadání a potvrzení hesla a umožňují skupiny uzamknout, pokud třeba chceme někomu ukázat náš program, ale určité části chceme skrýt.
4. Nyní klepneme levým tlačítkem myši na číslo horní podmínky a tím ji označíme.
5. Podržíme klávesu **Shift** a klepneme na číslo spodní podmínky, takže máme všechny označené jako na obrázku níže.
6. Přetáhneme označené podmínky na prázdný řádek **New condition** náležící pod vytvořenou skupinu.
7. Dvojitým poklepáním na název skupiny ji můžeme zavřít či otevřít, to ovšem nemá vliv na funkčnost a slouží jen pro zpřehlednění.



Obrázek 4.10: **Přesun podmínek do skupiny**

Skupinu máme vytvořenou a nyní založíme událost (již mimo skupinu – v té máme jen události týkající se manipulace s objekty!), kdy klepnutí na objekt pro start hry vyvolá spuštění pohybu kuličky a deaktivování skupiny s událostmi pro manipulace s objekty:

1. Vytvoříme novou podmínku z objektu **The mouse pointer and keyboard** ( ) a zvolíme **The mouse→User clicks on an object**.
2. V seznamu zvolíme objekt tlačítka pro start hry.
3. První akci přiřadíme pod objektem **Special** ( ), a to **Groups→Deactivate**.
4. Zvolíme název skupiny a potvrdíme.
5. Objektu kuličky přiřadíme akci **Movement→Start**.

Než budeme moci základní verzi ozkoušet, musíme vytvořit ještě události pro náraz do pozadí, odrazové plochy či okraje obrazovky.

1. Vytvoříme podmínku z objektu kuličky a zvolíme **Collisions→Another object**.
2. V seznamu vybereme symbol zástupce skupiny.
3. Kuličce přiřadíme akci **Movement→Bounce**.
4. Další podmínka z objektu kuličky bude **Collisions→Backdrop** a přiřadíme jí stejnou akci jako v kroku 3.
5. Poslední podmínku u objektu kuličky týkající se odrazu bude **Position→Test position of „Název objektu“**.
6. Označíme všechny 4 směry vycházející z obrazovky ven, potvrdíme a zopakujeme krok č. 3.
7. Nyní vytvoříme novou podmínku z objektu jamky a vybereme **Collisions→ Overlapping another object**.
8. V seznamu objektů vybereme kuličku.

9. Pod objektem **Storyboard controls** () přiřadíme **Next frame** pro pokračování dál či **Restart the current frame** pro opakování snímku.

Můžeme hru spustit a ozkoušet, zda je vše v pořádku. Ačkoli je hra již funkční a hratelná, my se budeme zabývat dalšími doplňujícími prvky.

Vliv prostředí

Jako první vložíme pro zpestření herní úrovně vodu. Přesněji se bude jednat o vodní proud, do kterého bude moci kulička spadnout. Při počáteční velké rychlosti sice dojde k jejímu zpomalení a vychýlení směru, ovšem dostane se na druhý břeh a bude pokračovat dál. Pokud se však dostane do vody podruhé, bude již příliš pomalá a proud ji unese.

Nejprve vložíme samotný objekt vody, který ani nebude animovaný, protože nám slouží jen pro demonstraci:

1. Vložíme aktivní objekt a použijeme grafiku ze souboru Zdroje\Logicka\logVoda.PNG.
2. V jeho **Properties** pod ikonou **Display Options** () nastavíme **Ink effect** na **Add**.

Poznámka: Operátor ADD způsobí, že hodnoty RGB komponent pixelů objektů jsou přičteny k hodnotám pixelů pozadí či objektu, který je překrýván.

3. Klepneme na objekt pravým tlačítkem myši a zvolíme **Order→To Back**.
4. Objekt nakopírujeme a umístíme podle obrázku 4.1.

Přepneme se do Event editoru a vytvoříme událost, kdy se kulička dotýká vody a je tak ovlivněn jeho pohyb:

1. Vytvoříme podmínku z objektu kuličky a zvolíme **Collisions→Overlapping another object**.
2. Zvolíme objekt vody a potvrdíme.
3. Objektu kuličky přiřadíme nejprve akci **Position→Set Y Coordinate**.
4. Klepneme na tlačítko **Retrieve data from an object** a z objektu kuličky vytáhneme **Position→Y Coordinate**.
5. Dopíšeme znaménko plus a číslo 2.5, takže výraz vypadá: `Y( "kulička" )+2.5`.
6. Druhá akce u objektu kuličky bude **Movement→Set speed**.
7. Klepneme na tlačítko **Retrieve data from an object** a tentokrát z kuličky vytáhneme **Movement→Speed**.
8. Dopíšeme znaménko minus a hodnotu 0.5, výraz má tedy tvar `Speed( "Active 4" )-0.5`

Tyto akce způsobí, že dostane-li se kulička do vody, začne ji proud unášet směrem dolů – k souřadnici Y přičítáme číslo 2.5, což tedy udává rychlost proudu. Zároveň ji voda po celou dobu kontaktu zpomaluje o hodnotu 0.5. Pokud je rychlost nulová, převáží pohyb po souřadnici Y a kuličku unáší proud.

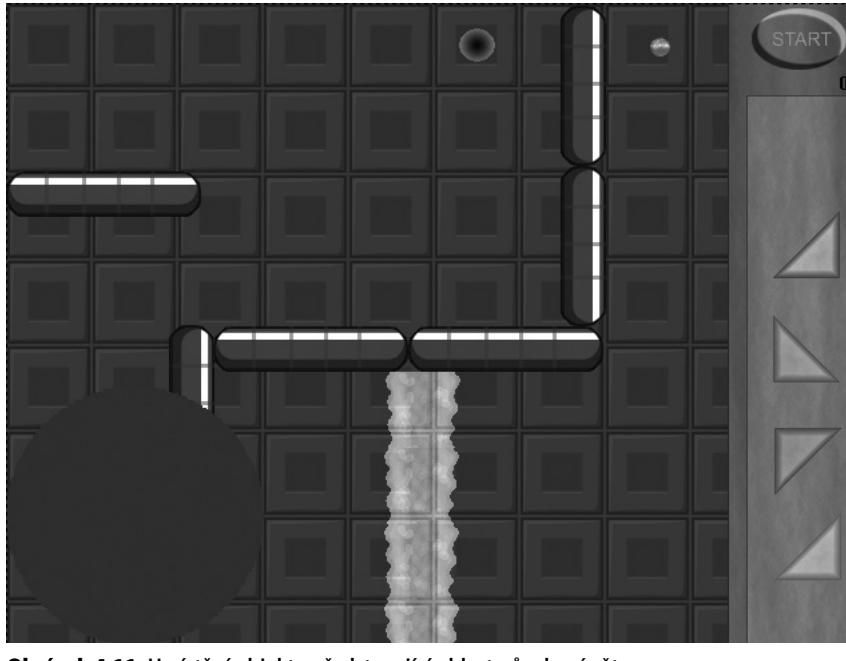
Druhým vlivem prostředí v naší hře bude vanoucí vzduch. Ten se bude objevovat v časových intervalech a bude vytvářen větrákem, abychom naznačili jeho přítomnost. Průvan pak kuličku vychýlí či postupně přetlačí směrem, kterým vane.

Tentokrát nebudeme používat žádné funkce a ukážeme si, jak lze celou věc vyřešit zábavně v rámci daných podmínek, akcí a pomocných objektů.

Nejprve vytvoříme objekt větráku a vytvoříme animaci ze souborů Zdroje\Logicka\logVetrak1.PNG a Zdroje\Logicka\logVetrak2.PNG a umístíme jej jako na obrázku 4.1.

Teď si vytvoříme objekt, který bude představovat oblast větru. V podstatě tedy vymezíme, kde má na kuličku vítr působit. Tento objekt bude během hry samozřejmě neviditelný.

1. Vložíme aktivní objekt a použijeme grafiku ze souboru Zdroje\Logicka\logVitr.PNG.
2. Tento objekt umístíme jako na obrázku níže.



Obrázek 4.11: Umístění objektu představující oblast působení větru

3. Zobrazíme si jeho Properties a pod ikonou **Display Options** ( ) vyškrtneme volbu **Display at start**.

Ještě než se pustíme do samotného působení větru na pohyb kuličky, nastavíme časové intervaly zapínaní větráku, a tím pádem i samotného průvanu.

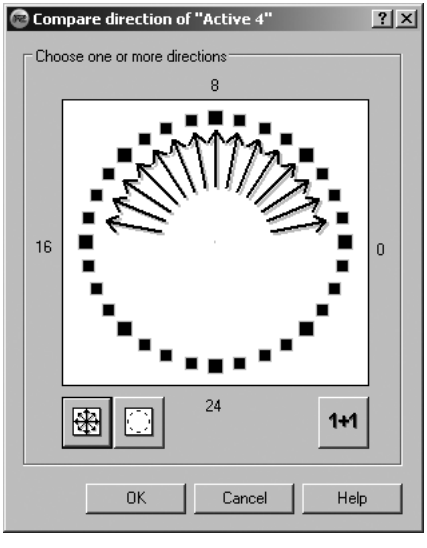
1. Přepneme se do **Event editoru** a vytvoříme novou podmínku z objektu **Timer** ( ), kde zvolíme **Every**.
2. Zadáme hodnotu v řádku **Second(s)** na 3 a potvrdíme.
3. U objektu představujícího oblast větru zvolíme akci **Flags→Toggle**.
4. Zadáme číslo 1 a potvrdíme.

Každé 3 sekundy se tedy hodnota vnitřního přepínače číslo 1 u daného objektu změní na opačnou. Pro oba stavy nyní definujeme, jak se má chovat větrák:

1. Vložíme novou podmínku z objektu oblasti větru a zvolíme **Alterable Values→Flags→Is a flag on?**.
2. Zadáme číslo 1.
3. Větráku přiřadíme akci **Animation→Start**.
4. Vytvoříme další podmínku z objektu oblasti větru a zvolíme **Alterable Values→Flags→Is a flag off?**.
5. Teď větráku přiřadíme akci **Animation→Stop**.

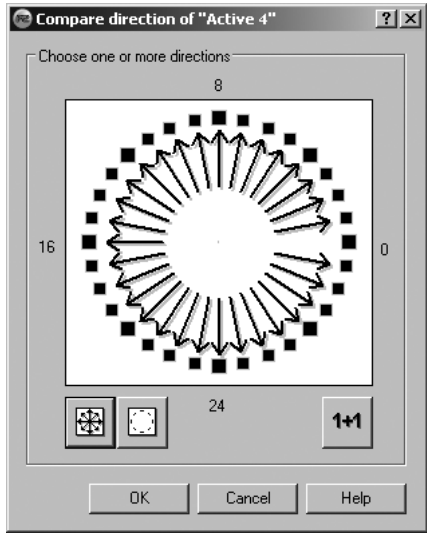
A nyní můžeme vytvořit potřebné události pro situaci, kdy má být pohyb kuličky ovlivňován působením větru. Podmínky tedy především zjišťují, zda je větrák zapnutý a kulička se nachází (překrývá ji) v oblasti působení větru.

1. Novou podmínku vytvoříme z objektu kuličky a zvolíme **Collisions→Overlapping another object**.



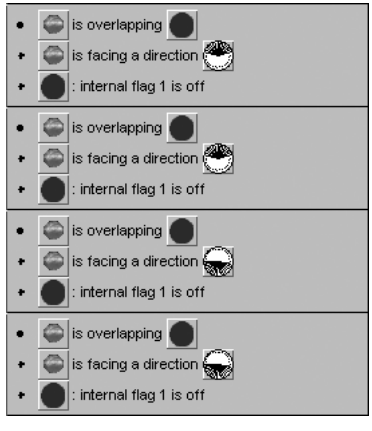
Obrázek 4.12:
Volba směrů 15 až 1

2. V seznamu objektů vybereme náš objekt představující oblast působení větru.
3. Vložíme vnořenou podmínku a z objektu kuličky vybereme **Direction→Compare direction of „Název objektu“**.
4. Vybereme všechny směry z horní poloviny kruhu kromě směrů vpravo a vlevo tak, jak je na obrázku 4.12.
6. Vložíme druhou vnořenou podmínku a tentokrát z objektu oblasti větru vybereme **Alterable Values→Flags→Is a flag on?**.
5. Vepíšeme číslo 1 a potvrdíme.



Obrázek 4.13:
Volba směrů 16 až 31

6. Celou podmínku zkopírujeme a v kopii poklepeme na vnořenou podmínku týkající se vymezení směrů.
7. Vybereme směry v dolní polovině kruhu včetně směru doleva a potvrdíme.



Obrázek 4.14:
Podmínky pro ovlivnění kuličky větrem

8. Obě podmínky označíme a zkopírujeme, celkem máme tedy nyní od každé z nich dvě kopie.

Dvě podmínky s odlišnou tolerancí směru kuličky jsme vytvořili proto, aby působení větru na ni záviselo na tom, jakým směrem se původně pohybovala. Pokud bychom měli jen jednu podmínku, kulička by se otáčela vždy jen doleva či doprava nezávisle na tom, jaký byl předtím její pohyb.

Zkopírováním podmínek jsme program donutili zpracovat akce dvakrát, a tím pádem bude i efekt působení dvojnásobný. Teď nastavíme akce, které způsobí samotnou změnu směru:

1. Nyní podmínce, která obsahuje požadavek na pohyb kuličky jedním ze směrů nahoru (), nastavíme akci **Direction→Select Direction**.
2. Klepneme na tlačítko **Calculate Direction** ().
3. Klepneme na tlačítko **Retrieve data from an object** a z objektu kuličky vytáhneme **Animation→Current direction value**.
4. Vepíšeme znaménko minus a číslo jedno, výraz tedy vypadá takto: `Dir( "Název objektu" )-1`.
5. Tuto akci přeneseme i do dalších 3 podmínek.
6. U těch, co však obsahují požadavek na pohyb kuličky jedním ze směrů dolů (), změníme ve výrazu znaménko minus na plus.

Máme tedy čtyři podmínky, z nichž první dvě mají nastaveno jako akci směr kuličky přičítat a zbylé dvě naopak odečítat.

Smyslem těchto událostí je plynulá změna směru kuličky. Vítr působí doprava, a proto působí-li na kuličku, vlastně ji otáčí, dokud není její směr stejný, protože vnořené podmínky neplatí pro směr vpravo.

Ještě bychom měli vytvořit událost, kdy při stisku klávesy Esc dojde k restartování snímku. Pokud to neuděláme, při neúspěšném pokusu dojde hra k slepému konci a hráči nezbude nic jiného, než ji vypnout.

Druhá úroveň

V druhé úrovni přidáme některé odlišné prvky jako zrychlovací pás či přemisťovací jamku, sloužící jako teleport, do které bude muset hráč pomocí vhodně umístěných odrazových ploch míček navést.

Přepneme se do **Storyboard editoru**, klepneme na druhý snímek pravým tlačítkem myši a zvolíme **Clone**, čímž jej zkopírujeme.

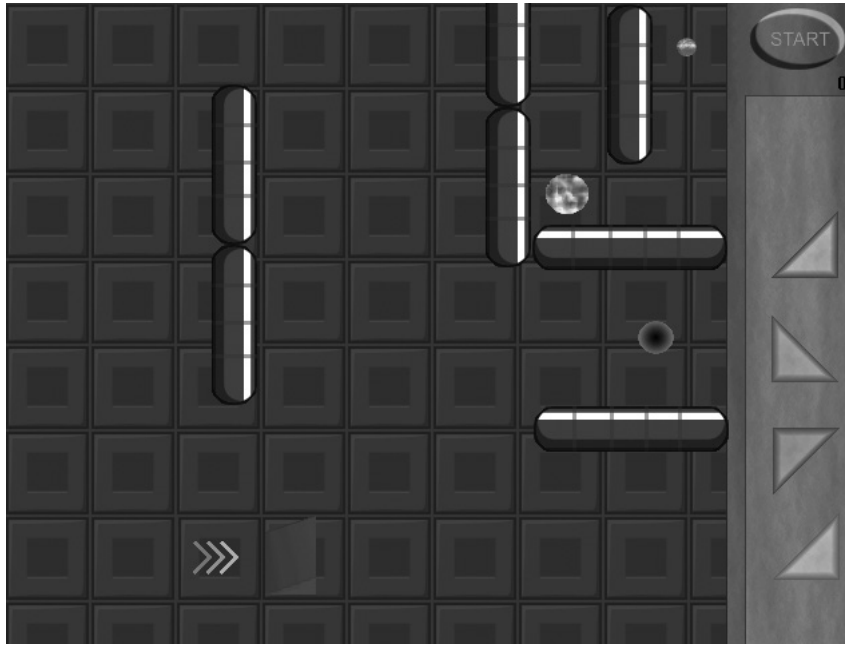
Otevřeme si jej pro úpravy a jako první smažeme objekty vody, větráku a větru. Při jejich smazání budeme programem dotázáni, zda mají být vymazány i podmínky k těmto objektům přiřazené, což potvrdíme.

Přepneme se do **Event editoru**, protože musíme vymazat zbylé události působení větru na kuličku. Z těch bylo odstraněny zmínky o již vymazaném objektu oblasti větru a podmínky tak vypadají jako na obrázku níže. Jejich ponechání by znemožnilo hru.

Obrázek 4.15
Nekompletní podmínky po vymazání zúčastněného objektu

Upravíme umístění překážek a cílové jamky tak, aby bylo jako na obrázku níže.

1. Vložíme aktivní objekt a použijeme grafiku ze souboru Zdroje\Logicka\logPort.PNG.
2. V jeho **Properties** pod **Display Options** ( ) nastavíme pod **Ink Effect** volbu **Add**.
3. Objekt teleportu umístíme podle obrázku níže.
4. Přepneme se do **Event editoru** a z objektu kuličky vytvoříme novou podmínku **Collisions→Overlapping another object**.
5. Ze seznamu vybereme objekt teleportu a potvrdíme.
6. Kuličce přiřadíme akci **Position→Select Position**.



Obrázek 4.16: Druhá úroveň

7. Ručně zadáme hodnotu X na 50 a Y na 80.

8. Dále objektu kuličky navolíme **Direction→Select Direction** a zvolíme pouze směr dolů.

Teď vytvoříme zrychlovací pás, který má samozřejmě za účel zvýšit rychlost pohybu kuličky. To ovšem má svůj důvod – v této úrovně, bude kulička postupně zpomalovat a tak pokud hráč tento pás nevyužije, nedostane kuličku do cíle.

1. Zobrazíme si **Properties** objektu kuličky a pod ikonou **Movement** () změníme hodnotu **Deceleration** na 8.
2. Vytvoříme tedy nový aktivní objekt a jeho grafiku vložíme ze souboru Zdroje\Logicka\logPas.PNG.
3. Zobrazíme si jeho **Properties** a pod ikonou **Runtime Options** () odškrtneme volbu **Use fine detection**, aby program počítal s celou plochou objektu.
4. Na objekt nyní klepneme pravým tlačítkem myši a zvolíme **Order→To Back**, jinak by překrýval kuličku.
5. Pás umístíme podobně jako obrázku 4.16.
6. V Event editoru vytvoříme podmínku z objektu kuličky, a to **Collisions→Overlapping another object**, kde zvolíme objekt pásu.
7. Kuličce přiřadíme akci **Movement→Set Speed**.
8. Klepneme na tlačítko **Retrieve data from an object** a z objektu kuličky vytáhneme **Movement→Speed**.
9. Dopíšeme výraz tak, aby měl následující tvar: `Min(Speed( "Název objektu" )+2, 65)`.

Použili jsme funkci menší hodnoty, která ze dvou zadaných čísel použije vždy menší z nich. V tomto případě to slouží k tomu, že rychlost sice poroste o 2, ale nemůže být vyšší než 65. Použitím této funkce jsme tedy omezili růst rychlosti, která by nad uvedené číslo byla již příliš velká.

Poznámka: Stejně tak existuje funkce Max (první hodnota, druhá hodnota), která naopak vrací vždy vyšší z obou hodnot.

Zajímavým a poměrně složitým prvkem bude vyskočení kuličky po nájezdu na jakýsi skokanský můstek, umístěný za zrychlovacím pásem. Skok bude skutečně závislý na rychlosti a k jeho výpočtu budeme potřebovat funkci sinus.

1. Nejprve vložíme aktivní objekt a vložíme grafiku ze souboru Zdroje\Logicka\logMustek.PNG.
2. Objekt umístíme za zrychlovací pás jako na obrázku 4.16.
3. Klepneme na objekt pravým tlačítkem myši a zvolíme **Order→To Back**.
4. Z tohoto objektu vytvoříme nyní podmínku **Collisions→Another Object**.
5. Vybereme objekt kuličky a potvrdíme.

6. Kuličce přiřadíme akci **Alterable Values→Set**.
7. Zvolíme **Alterable Value C** a vepíšeme číslo 1.
8. Vytvoříme ještě jednu akci, stejnou jako v kroku 6.
9. Zvolíme **Alterable Value Y** a klepneme na tlačítko **Retrieve data from an object** a z objektu kuličky vytáhneme **Position→Y Coordinate**.

Tato událost tedy způsobí, že po doteku kuličky se „skokanským můstkem" se její proměnná C nastaví na 1 a proměnná Y na aktuální hodnotu souřadnice Y. Pokud je tedy C=1, pak je objekt ve stavu skoku, který nyní definujeme:

1. Vytvoříme novou podmínku z objektu kuličky a vybereme **Alterable Values→ Compare to one of the alterable values**.
2. Zvolíme **Alterable Value C** a vepíšeme číslo 1.
3. Objektu kuličky přiřadíme akci **Alterable Values→Add to**.
4. Klepneme na tlačítko **Retrieve data from an object** a z objektu kuličky vybereme **Values→Values A to M→Retrieve Alterable Value A**.
5. Nyní výraz upravíme tak, aby vypadal takto: `Max(Alterable Value A( "Název objektu" )-0.75, -8)`.

Tato akce tedy stále snižuje hodnotu proměnné A o číslo stále se zvyšující o 0.75 až do hodnoty -8. Tuto rostoucí proměnnou zahrneme do výpočtu v následující akci:

1. Ve stejné podmínce přiřadíme kuličce druhou akci, a to **Position→Y Coordinate**.
2. Klepneme na tlačítko **Retrieve data from an object** a u objektu kuličky vybereme **Values→Values N to Y→Retrieve Alterable Value Y**.
3. Vepíšeme znaménko plus a klepneme na tlačítko **Sin**.
4. Klepneme opět na tlačítko **Retrieve data from an object** a u objektu kuličky vybereme **Values→Values A to M→Retrieve Alterable Value A**.
5. Za závorku vepíšeme znak násobení „*" a naposledy klepneme na **Retrieve data from an object**.
6. Z objektu kuličky vytáhneme **Movement→Speed**.
7. Dopíšeme znak pro dělení „/" a číslo 2, takže celý výraz bude vypadat takto: `Alterable Value Y( "mic" )+Sin(Alterable Value A( "mic" ))*Speed( "mic" )/2`.

Nyní si tuto komplikovanou událost zkusíme trochu přiblížit, abychom pochopili její fungování.

Jestliže je hodnota C=1 a objekt se tak nachází ve stavu skoku, jeho proměnná A klesá o stále větší číslo – na konci. Zároveň se pozice Y vypočítává tak, že k uložené hodnotě souřadnice Y (která se zaznamenala při doteku se skokanským můstkem) se přičítá výsledek funkce sinus (Hodnota A) násobený poloviční rychlostí míčku.

Funkce sinus vrací výsledek od -1 do 1. Díky snižování proměnné A stále rostoucím číslem bude průběh s časem funkce rychlejší a skok kuličky skutečnější.

Protože změnu souřadnice Y o čísla -1 až 1 bychom ani nezpozorovali, násobíme je poloviční hodnotou rychlosti kuličky. Z čehož je zřejmé, že čím vyšší bude rychlost, tím větší skok nastane, protože násobíme vyšším číslem. Dělení dvěma je čistě pro snížení výsledku, který by byl pro tyto účely moc vysoký.

Jako poslední musíme zajistit, aby skok skutečně vypadal jako skok a objekt tak opsal jakýsi půlkruh. Proto musí funkce probíhat pouze do poloviny – tedy počítat maximálně s hodnotou nad -180 (360/2=180).

1. Vytvoříme novou podmínku z objektu kuličky a zvolíme **Alterable Values→ Compare to one of the alterable value**.
2. Zvolíme **Alterable Value A** a operátor změníme na **Lower or equal**.
3. Vepíšeme -180 a potvrdíme.
4. Kuličce přiřadíme nejprve **Alterable Values→Set** a proměnnou A potvrdíme s nulovou hodnotou.
5. Poté přiřadíme stejnou akci jako v předchozím kroku, vynulujeme ovšem proměnnou C a ukončíme tak stav skoku.

Pokud tedy proměnná A objektu kuličky klesne pod -180, musí dojít k ukončení skoku, protože 180° je polovina kruhu, a tím pádem musíme dalšímu zvyšování této hodnoty zabránit. Nastavíme tedy hodnotu proměnné C na 0, a tím deaktivujeme událost pro zvyšování proměnné A. Také tu vynulujeme.

Mohli bychom samozřejmě vymyslet i další a další prvky, ovšem to již nemá smysl dělat společně. Dokončili jsme tak úspěšně další hru a můžeme pokračovat další kapitolou, která bude trochu odlišná, rozhodně však zajímavá.