Kurz LSL skriptování

Shiny Iceberg

2009

součást cyklu
Nejsme jelita

Kurz LSL skriptování

Shiny Iceberg
v Second Life od roku 2006
shiny.iceberg@virtualmagazine.cz

Aktuální projekty

virtualmagazine.cz
Urbanica, Shinyland
Bwindi Orphans
cyklus Nejsme jelita

Plán přednášky

1. Fungování a struktura skriptu
2. Z čeho se skládá skript
3. Vlastnosti objektu
4. Pohyb objektu
5. Pose bally
6. Komunikace skriptu
7. Inventory objektu
8. Detektory
9. Particles
10. Příklady a dokumentace

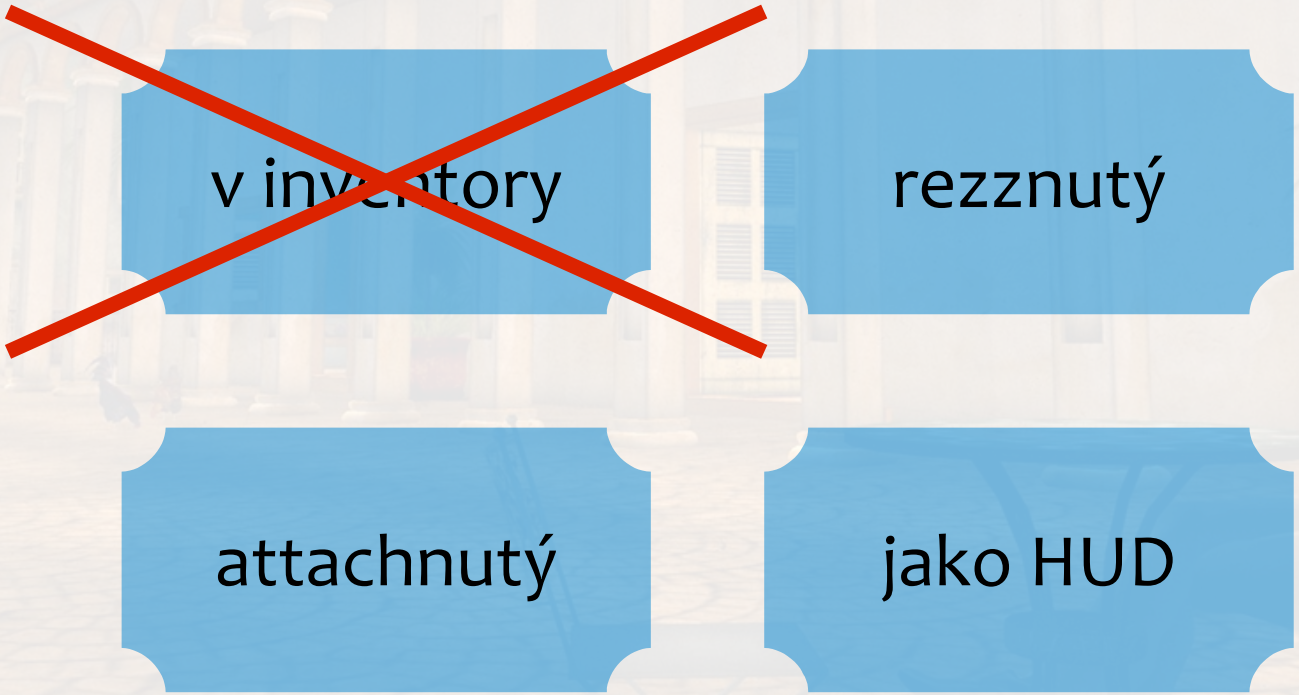
Stav objektu

Každý objekt může být v několika stavech

- v inventory
- rezznutý
- attachnutý
- jako HUD

Stav objektu

Skript v inventory je pozastaven a nereaguje

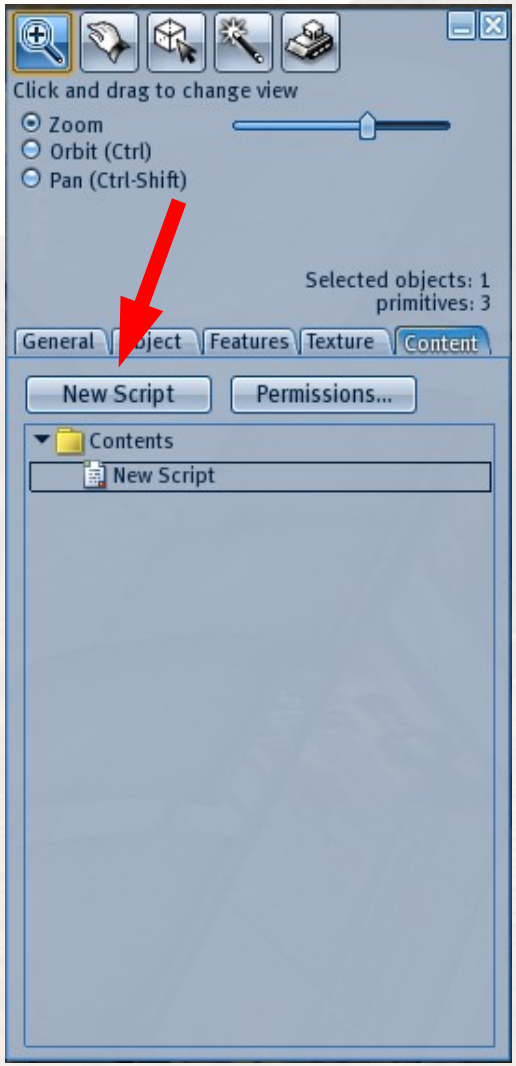


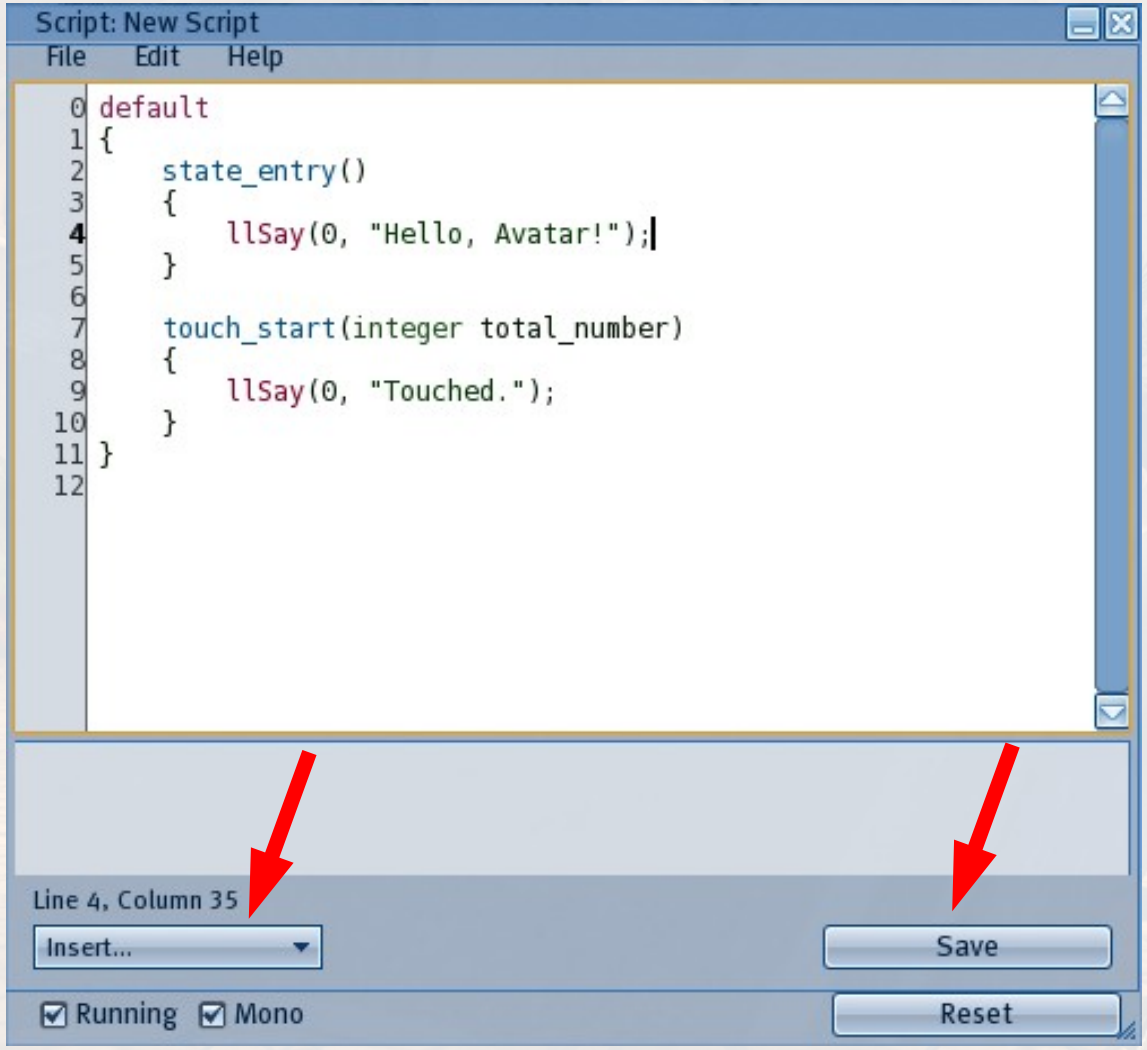
Jak funguje skript

Skript v Second Life

- dokáže pracovat pouze v objektu a pracuje i při vašem odhlášení
- umožňuje interakci s objekty, avatary, ostatními skripty, virtuálním prostředím či internetem
- funguje na principu podnět → reakce neboli „co se bude dít, když ...“

Co potřebujete do začátku





Láska na první pohled

```
default
{
    state_entry()
    {
        llSay(0, "Dobry den!");
    }

    touch_start(integer total_number)
    {
        string Text = "Kliknul jsi" + " na mne.";

        if ( SQRT2 > 2 )
        {
            llSay(0, Text );
        }
    }
}
```

1. STATES
2. EVENTS
3. VARIABLES
4. CONSTANTS
5. FLOW CONTROL
6. FUNCTIONS
7. OPERATORS

Z čeho se skript skládá

Každý **skript** obsahuje několik základních elementů

- STATES – stav
- EVENTS – událost
- VARIABLES – proměnné různých typů
- CONSTANTS – konstanty
- FLOW CONTROL – řízení běhu skriptu
- FUNCTIONS – definované kusy kódu
- OPERATORS – znaky operace

Hlavní znaky kódu

- Kód tvoří bloky příkazů uzavřené mezi { a }
- Každý řádek s příkazem musí končit znakem ;
- Text napsaný za // je ignorován jako komentář
- Parametry funkcí se uvádí mezi (a) a oddělují se čárkou
- Interní SL editor pomáhá barevně odlišovat

States – stavy objektu

STATES

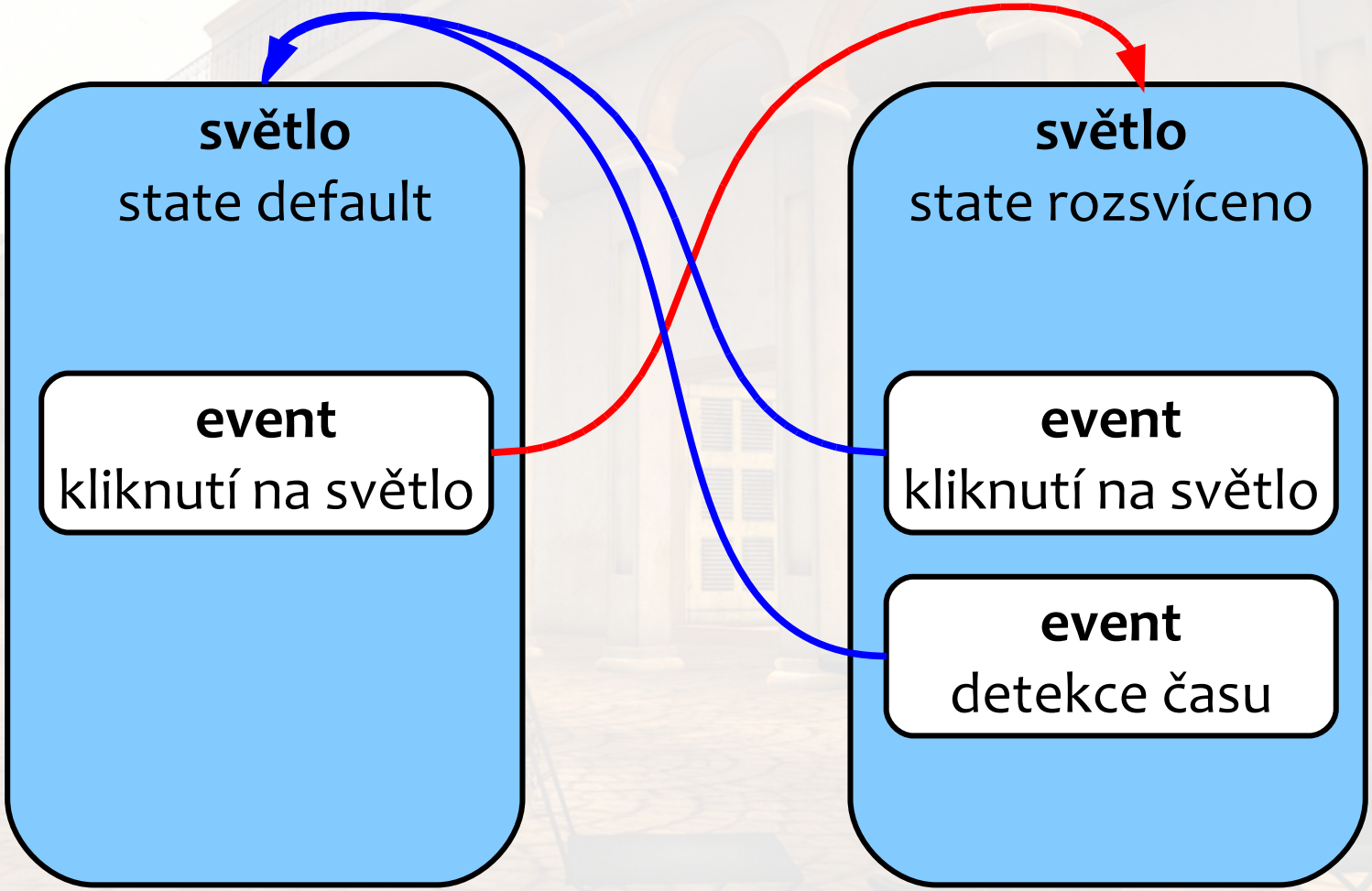
- jsou různé stavy či situace skriptu
- každý skript musí obsahovat alespoň *state* DEFAULT, který je aktivován vždy jako první při restartu
- příkladem jsou třeba
  - dveře: default, otevřené, zamčené (3 states)
  - světlo: default, rozsvíceno (2 states)
  - teleport: default (1 state)

Events – nastalé situace

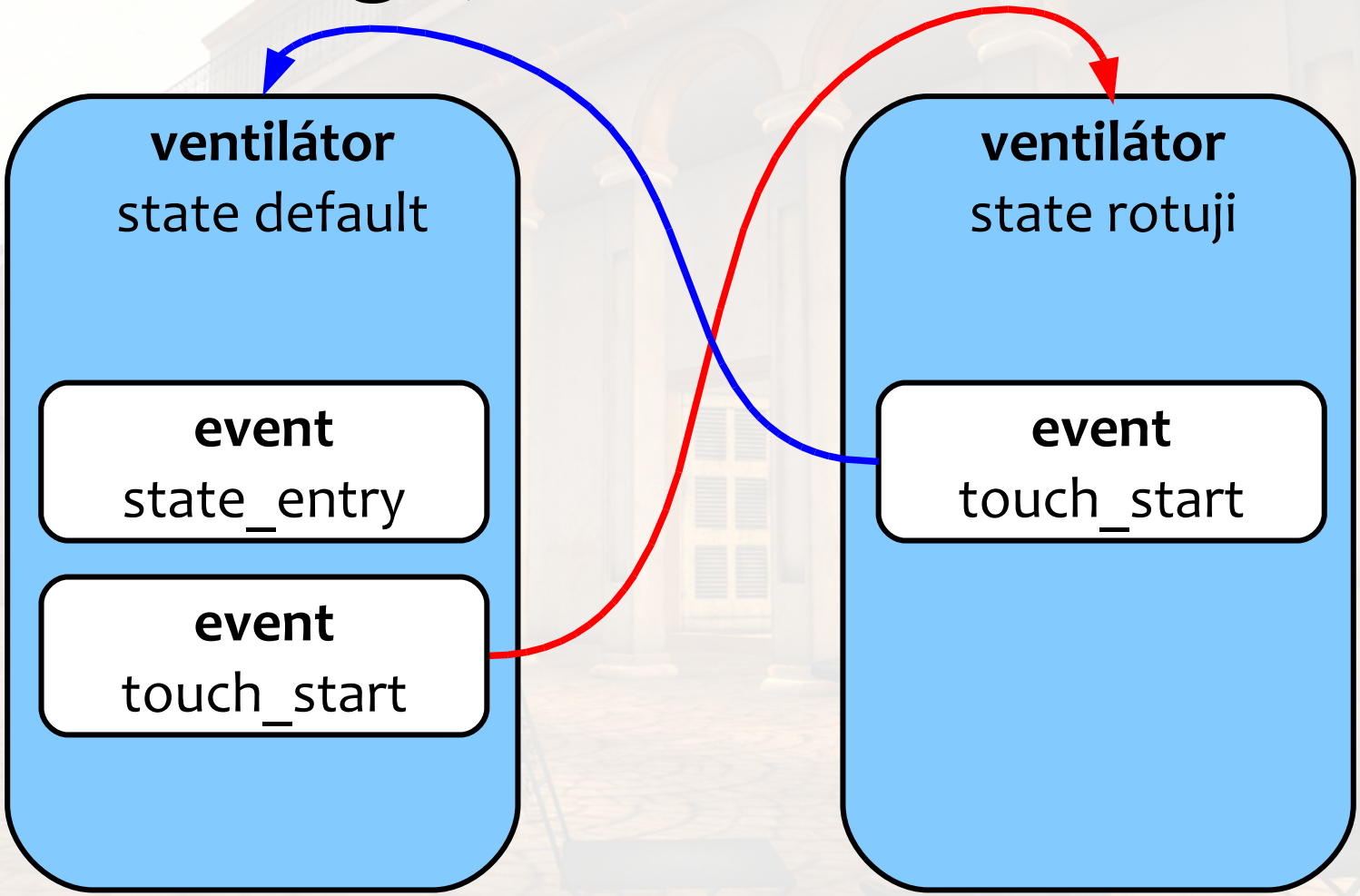
EVENTS

- jsou události, které nastaly a které vyžadují reakci
- každý *state* musí obsahovat alespoň 1 *event*
- příkladem jsou třeba
  - dveře: kliknutí na dveře, uplynutí určitého času, příchod ke dveřím
  - světlo: kliknutí na světlo, detekce času
  - teleport: sednutí na teleport, kliknutí na teleport

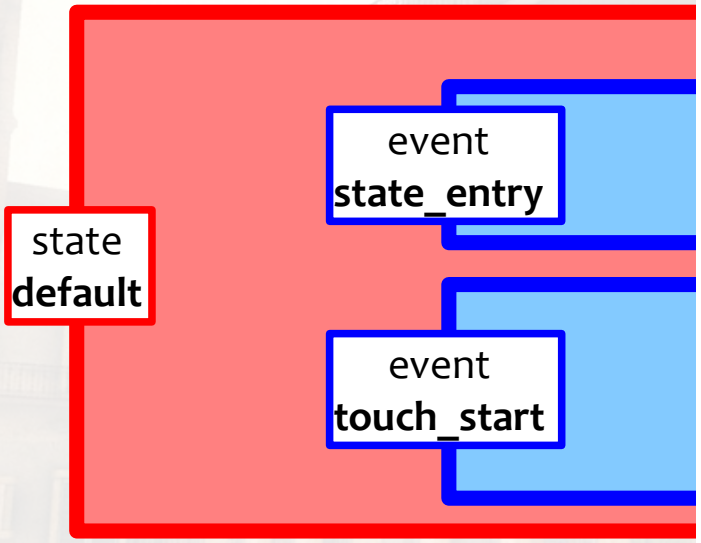
Jak funguje State a Event 1/2



Jak funguje State a Event 2/2

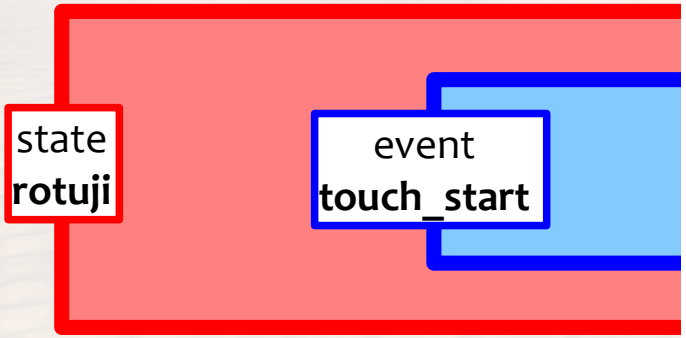


Uspořádání States a Events




```
default
{
    state_entry()
    {
        llSay(0, "Po kliknutí budu rotovat");
    }

    touch_start(integer total_number)
    {
        // Prikaz na rotovani objektu
        llTargetOmega( <0,0,1>, 1, 1);
        state rotuji;
    }
}
```




```
state rotuji
{
    touch_start(integer total_number)
    {
        llTargetOmega( <0,0,1>, 1, 0);
        state default;
    }

}
```

Nejdůležitější Events

attach – objekt byl obléknut či svlečen
changed – když se změní vlastnosti objektu
link_message – skripty v primsetu si posílají zprávy
listen – odchytávání textu z chatu
money – když objektu někdo zaplatí
on_rez – při vyndání objektu z inventory
state entry – při přechodu do *state*
timer – po uplynutí určitého času
touch_start – na objekt někdo kliknul myší

Kompletní seznam: *http://lslwiki.net/lslwiki/wakka.php?wakka=events*

Nejdůležitější Events

attach(key *attached*)

{ <příkaz1>; <příkaz2>; … }

- *attached* obsahuje identifikátor nositele

Nejdůležitější Events

changed(integer *change*)

{ <příkaz1>; <příkaz2>; … }

- *change* obsahuje, co se na objektu změnilo

CHANGED_INVENTORY	změna inventory objektu
CHANGED_COLOR	změna barvy / průhlednosti
CHANGED_SHAPE	změna tvaru objektu (ořezání, typ primu)
CHANGED_SCALE	změna velikosti objektu
CHANGED_TEXTURE	změna textury objektu, glow, rotace textury
CHANGED_LINK	změna slinkování, posazení avatara
CHANGED_ALLOWED_DROP	změna povolení vhazovat do objektu věci
CHANGED_OWNER	změna vlastníka objektu
CHANGED_REGION	změna regionu, kde je objekt
CHANGED_TELEPORT	pokud byl objekt teleportován

Nejdůležitější Events

link_message(integer *sender_num*, integer *num*, string *str*, key *id*)
{ <příkaz1>; <příkaz2>; … }

- Odchytává zprávy zaslané pomocí *llMessageLinked*
- *sender_num* obsahuje číslo primu v primsetu, který zprávu poslal
- 3 další proměnné obsahují číslo, řetězec a identifikátor předaný zprávou

Nejdůležitější Events

listen(integer *channel*, string *name*, key *id*, string *message*)
{ <příkaz1>; <příkaz2>; … }

- Odchytává zprávy z chatu i skrytých kanálů, definuje se pomocí funkce *llListen*
- *channel* obsahuje číslo kanálu, na kterém se zpráva zachytila
- *name* a *id* jsou jméno a identifikátor autora zprávy
- *message* je text odchycené zprávy

Nejdůležitější Events

money(key *id*, integer *amount*)
{ <příkaz1>; <příkaz2>; … }

- *id* obsahuje identifikátor nositele
- *amount* obsahuje výši darované částky

Nejdůležitější Events

on_rez(integer *start_param*)
{ <příkaz1>; <příkaz2>; … }

- *start_param* obsahuje speciální parametr při rezzování objektu pomocí skriptu, při vyndání z inventory je vždy nula

Nejdůležitější Events

state_entry()

{ <příkaz1>; <příkaz2>; … }

- zcela bez parametrů

Nejdůležitější Events

timer()

{ <příkaz1>; <příkaz2>; … }

- zcela bez parametrů
- nastává po čase určeném pomocí *llSetTimerEvent*

Nejdůležitější Events

touch_start(integer *num_detected*)
{ <příkaz1>; <příkaz2>; … }

- *num_detected* obsahuje počet detekovaných agentů, obvykle se pracuje s agentem s číslem nula, třeba *llDetectedKey(0)*

Kde najdu příklady

Anglicky

- www.lslwiki.net/lslwiki
- www.2lifeblog.com/content/blogsection/13/72/
- xahlee.org/sl/ls.html
- www.3greeneggs.com/autoscript/

Česky

- vhelp.cz
- druhyzivot.cz

Otázky a diskuze