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

Organizační body

Průběh lekce

- bude trvat zhruba 90-120 minut
- pokud nekladete dotazy, vypněte si mikrofon
- příklady si klidně zkoušejte přímo v hledišti

Vaše otázky

- můžete se ptát na konci každého snímku
  nebo na konci celé přednášky
- dotazy mohou být přes voice nebo IM
- při psaní IM z posledních řad použijte Shout

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

Funkce a eventy pro komunikaci

- llOwnerSay
- llInstantMessage
- llSay / llWhisper / llShout
- llRegionSay
- llListen + event listen
- llDialog
- llMessageLinked + event link_message

```
llOwnerSay(string Text);
```

VSTUP: až 1023 znaků, které se mají odeslat vlastníkovi objektu

VÝSTUP: nic

POZNÁMKA:

- vlastník objektu musí být ve stejném regionu
- používá se primárně pro odladění skriptu nebo pro attachmenty

```
llInstantMessage(key User, string Text);
```

VSTUP:

- ***User:*** UUID avatara, kterému se má poslat zpráva
- ***Text:*** až 1023 znaků, které se mají poslat

VÝSTUP: nic

POZNÁMKA:

- zpráva se zobrazí jen cílovému avatarovi, případně je přeposlána na e-mail
- příkaz pozdrží skript na 2 sekundy (anti-spam)

```
llSay(integer Kanal, string Text);
llWhisper(integer Kanal, string Text);
llShout(integer Kanal, string Text);
```

VSTUP:

- ***Kanal:*** číslo kanálu, kam se má text poslat
  (-2 147 483 648 až 2 147 483 647, 0 = veřejný chat)
- ***Text:*** až 1023 znaků, které se mají poslat

POZNÁMKA:

- dosah - llWhisper 10 metrů, llSay 20 metrů, llShout 100 metrů
- lze použít symbol /*me*, který je ve veřejném chatu nahrazen jménem objektu

```
llRegionSay(integer Kanal, string Text);
```

VSTUP:

- ***Kanal:*** číslo kanálu, kam se má text poslat, nelze použít kanál 0 (veřejný chat)
- ***Text:*** až 1023 znaků, které se mají poslat

VÝSTUP: nic

POZNÁMKA:

- ideální pro komunikaci či synchronizaci různých skriptů v rámci regionu

```
integer llListen(integer Kanal, string
    Jmeno,key Id, string Zprava);
```

VSTUP:

- ***Kanal:*** číslo kanálu, na kterém se má monitorovat
- ***Jmeno:*** omezení na objekty/avatara určitého jména
- ***Id:*** omezení na objekt/avatara s určitým UUID
- ***Zprava:*** omezení jen na určitý text v chatu

VÝSTUP: identifikátor filtru, který lze později měnit a vypínat, často se ale tato hodnota ignoruje

Poznámky k llListen

- funkce nastaví monitorování chatu dle zadaného filtru, maximálně lze aktivovat 65 filtrů
- nefiltrované monitorování na veřejném chatu výrazně zatěžuje region
- při prodeji výrobků s llListen se doporučuje reset skriptu při změně vlastníka

Příklady k llListen

```
// Zcela otevřený filtr na veřejném chatu
llListen( 0, "", NULL_KEY, "" );

// Filtr na kanalu 1, ktery reaguje jen
// na zpravy od objektu se jmenem Ovladaci panel,
// techto objektu muze byt nekolik
llListen( 1, "Ovladaci panel", NULL_KEY, "" );

// Filtr verejneho chatu, ktery reaguje pouze
// na zpravy od vlastnika objektu se skriptem
// a ktery reaguje pouze na text zpravy LOL
llListen( 0, "", llGetOwner(), "LOL" );
```

event *listen*

```
listen(integer Kanal, string Jmeno,
   key Id, string Zprava);
```

Do proměnných se uloží následující hodnoty

- ***Kanál:*** číslo kanálu, ze kterého se odchytila zpráva
- ***Jmeno:*** jméno objektu/avatara, od koho přišla zpráva
- ***Id:*** UUID objektu/avatara, od koho přišla zpráva
- ***Zprava:*** samotný text odchycené zprávy

Příklady k eventu *listen*

```
default
{
    state_entry()
    {
        llListen( 0, "", llGetOwner(), "" );
    }

    listen( integer Kanal, string Jmeno, key Id, string Text )
    {
        if ((Text == "LOL") || (Text == "lol"))
        {
            llOwnerSay("Pozor na Scalexe!");
        }
    }
}
```

```
llDialog(key Avatar, string Zprava,
   list Tlacitka, integer Kanal)
```

VSTUP:

- ***Avatar:*** UUID avatara, kterému se dialog zobrazí
- ***Zprava:*** až 512 znaků textu, který se zobrazí
- ***Tlacitka:*** seznam popisků k tlačítkům, max. 12
- ***Kanal:*** kam se má poslat jméno stisknutého tlačítka

POZNÁMKA:

- prázdný seznam popisků tlačítek zobrazí okno se zprávou a tlačítkem OK
- vždy se objeví tlačítko Ignore (bez odpovědi)
- tlačítka se umisťují zleva doprava a odspodu

Příklady k *llDialog*

```
string Zprava = "Ahoj, máš me rád?";
list Tlacitka = ["Ano", "Ne", "Nevim"];

default
{
    state_entry()
    {
        llListen(8, "", NULL_KEY, "");
    }

    touch_start(integer cislo)
    {
        // zjisti UUID avatara, ktery klikl na objekt
        key KdoToKlepe = llDetectedKey(0);
        llDialog(KdoToKlepe, Zprava, Tlacitka, 8);
    }

    listen(integer Kanal, string Jmeno, key Avatar, string Text)
    {
        if (Text == "Ano")
            llInstantMessage(Avatar, "Já te mám taky rád.");
        if (Text == "Ne")
            llInstantMessage(Avatar, "Fajn, já tebe taky ne!");
    }
}
```

llMessageLinked(integer *PoradiPrimu*, integer *Cislo*, string *Text*, key *ID*)

VSTUP:

- ***PoradiPrimu:*** pořadové číslo cílového primu v rámci linksetu (root má číslo 1, dále podle slinkování)
- ***Cislo, Text, ID:*** hodnoty předávané do dalšího primu

POZNÁMKA:

- nejvhodnější na komunikaci skriptů v jednom objektu - skrytá komunikace bez zpoždění
- pro číslo primu lze použít konstanty

LINK_SET	všechny primy
LINK_ALL_OTHERS	všechny ostatní primy
LINK_ALL_CHILDREN	všechny primu kromě root primu
LINK_THIS	*poslat* sám na sebe
LINK_ROOT	root prim

event *link_message*

```
link_message(integer PoradiPrimu,
   integer Cislo, string Text, key ID);
```

Do proměnných se uloží následující hodnoty

- ***PoradiPrimu:*** pořadové číslo primu v rámci linksetu, který poslal zprávu
- ***Cislo, Text, ID:*** hodnoty předané z původního primu

Příklady k *link_message*

```
// Tento skript vlozit do jednoho primu v linksetu
default
{
    touch_start(integer cislo)
    {
        llMessageLinked(LINK_SET, 1500, "Posilam penize", NULL_KEY);
        llSleep(5.0);
        llMessageLinked(LINK_SET, 2000, "Vrat s urokem", NULL_KEY);
    }
}
```

```
// Tento skript vlozit do druheho primu v linksetu
default
{
    link_message(integer prim, integer castka, string text, key id)
    {
        string penize = (string) castka;
        if (text == "Posilam penize")
            llOwnerSay("Hurá, dostal jsem " + penize + " korun.");
        if (text == "Vrat s urokem")
            llOwnerSay("Prý mám vrátit " + penize + " korun. Tuhle!");
    }
}
```

Pokročilé metody komunikace

Kromě již probraných způsobů komunikace existuje řada dalších pro některé speciální situace:

- funkce *llEmail* a event *email* slouží pro vzdálenou komunikaci mezi regiony, případně poslání/obdržení běžného e-mailu
- SL HTTP klient - umožňuje poslat skriptem HTTP dotaz na internet/server a obdržet odpověď
- SL HTTP server - skript odpovídá na HTTP dotazy z jiných skriptů nebo z internetu
- sada funkcí pro práci s XML-RPC (přenos dat)

Pokročilé metody komunikace

O těchto pokročilých způsobech komunikace se můžete dozvědět více:

- E-mail:
  http://www.lslwiki.net/lslwiki/wakka.php?wakka=Email
- SL HTTP klient a SL HTTP server
  http://wiki.secondlife.com/wiki/Category:LSL_HTTP
  http://wiki.secondlife.com/wiki/LSL_http_server
- XML-RPC
  http://www.lslwiki.net/lslwiki/wakka.php?wakka=XMLRPC
- Příklad v češtině na komunikaci SL a MySQL
  http://vhelp.cz/viewtopic.php?f=21&t=202

Příklad: Změna barvy nábytku 1/3

```
integer kanal;
key avatar;

default
{
    state_entry()
    {
        // vybere nahodny komunikacni kanal v rozmezi 1-1000
        // tento kanal slouzi pro komunikaci s uzivatelem
        kanal = (integer) llFrand(1000) + 1;
        llSitTarget ( <0.3, 0.0, 0.4>, <0.0, 0.0, 0.0, 1.0>);
        llListen(kanal, "", NULL_KEY, "");
    }

    touch_start(integer cislo)
    {
        avatar = llDetectedKey(0);
        llDialog(avatar, "Co chces zmenit?", ["Kostra", "Potah"], kanal);
    }
```

Příklad: Změna barvy nábytku 2/3

```
    listen(integer odkud, string jmeno, key id, string volba)
    {
        if (volba == "Kostra")
            llDialog(avatar, "Jakou barvu kostry?", ["Svetla", "Tmava"], kanal);
        if (volba == "Potah")
            llDialog(avatar, "Jakou barvu potahu?", ["Cerveny", "Modry", "Zeleny"], kanal);

        if (volba == "Svetla")
            llMessageLinked(LINK_SET, 1, "<1.0,1.0,1.0>", NULL_KEY);
        if (volba == "Tmava")
            llMessageLinked(LINK_SET, 1, "<0.2,0.2,0.2>", NULL_KEY);

        if (volba == "Cerveny")
            llMessageLinked(LINK_SET, 2, "<1.0,0.5,0.5>", NULL_KEY);
        if (volba == "Modry")
            llMessageLinked(LINK_SET, 2, "<0.5,0.5,1.0>", NULL_KEY);
        if (volba == "Zeleny")
            llMessageLinked(LINK_SET, 2, "<0.5,1.0,0.5>", NULL_KEY);
    }
}
```

Příklad: Změna barvy nábytku 3/3

```
// ----- Podrizeny skript pro KOSTRU -----
default
{
    link_message(integer prim, integer typ, string barvaText, key id)
    {
        if (typ == 1)
        {
            vector barvaVector = (vector) barvaText;
            llSetColor(barvaVector, ALL_SIDES);
        }
    }
}

// ----- Podrizeny skript pro POTAH -----
default
{
    link_message(integer prim, integer typ, string barvaText, key id)
    {
        if (typ == 2)
        {
            vector barvaVector = (vector) barvaText;
            llSetColor(barvaVector, ALL_SIDES);
        }
    }
}
```

Otázky a diskuze