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

Jak správně skriptovat

- cílem začínajícího skriptera je pochopit existující kód a upravit jej tak, aby dělal přibližně to, co potřebuje, případně sesmolit svůj kód, který funguje a on se tomu diví
- cílem pokročilého skriptera je napsat skript, který má veškeré potřebné funkce, ověřuje vstupy uživatele, běží rychle, nabízí uživatelský komfort, má čitelný a pochopitelný kód, šetří využívanou pamětí a minimálně zatěžuje simulátor

Čitelnost skriptů

- odsazujte vnořené kusy kódu pomocí tabelátoru
- pojmenujte vhodně proměnné a konstanty
- jasně označte názvem rozdíl mezi lokálními a globálními proměnnými a mezi konstantami
- používejte komentáře - ani málo, ani moc - tak, abyste se v kódu vyznali i po měsíci
- zalamujte dlouhé řádky kvůli lepší čitelnosti

Dobře a špatně čitelný příklad

```
// === Dobre citelny priklad ================
string INFOTEXT = "Kliknul jsi na mne.";

default
{
    state_entry()
    {
        llOwnerSay("Skript je připraven");
    }

    touch_start(integer pocetKliknuti)
    {
        // Poslu INFOTEXT do verejneho chatu
        llOwnerSay(0, INFOTEXT );
    }
}
```

```
// === Spatne citelny priklad ===============
string t="Kliknul jsi na mne.";
default{
state_entry(){
llOwnerSay("Skript je připraven");}
touch_start(integer i){
llOwnerSay(0,t);}}
```

Rychlost / paměťová náročnost

Existují metody pro extrémní optimalizaci skriptů, které se hodí pro speciální situace, nicméně pro obvyklou práci si stačí osvojit pár základních tipů, mezi které patří

- minimalizujte počet volání stejné funkce, pokud lze výsledek funkce uložit do proměnné a tu používat
- používejte primárně WHILE cyklus, je úspornější než FOR nebo DO...WHILE

Rychlost / paměťová náročnost

- minimalizujte počet použitých proměnných, využívejte vnoření funkcí (výstup jedné funkce je vstupem funkce druhé)
- rozdělte mega skripty do několika skriptů, které spolu komunikují
- speciální pozornost a testování vyžaduje práce s physical objekty a llListen / llSensorRepeat, bývají častým zdrojem lagování

Rychlost / paměťová náročnost

- proměnné list s větším počtem položek mohou zcela vyčerpat dostupnou paměť, např. list se seznamem 100 jmen avatarů má v průměru 5 kB (30% veškeré paměti pro skript)
- dlouhé cykly ve spojení s nastavením parametrů objektu generují lag
- výkonnostně náročné skripty pro vizuální změny objektů na vašem pozemku mohou být nastaveny tak, že běží jen v případě, že je poblíž nějaký avatar

Jeden problém, několik řešení

Cyklická změna barvy objektu

- obvykle řešeno pomocí *timer* a *llSetColor*
- co takhle použít barevnou texturu se šachovnicovou strukturou a funkci *llSetTextureAnim*

Rotace objektu

- obvykle řešeno pomocí cyklu a *llSetRot*
- co takhle použít *llTargetOmega*

Detekce avatara v blízkosti

- obvykle řešeno pomocí *llSensorRepeat*
- co takhle použít *llVolumeDetect*

Jeden problém, několik řešení

Hledání jména avatara v seznamu

- obvykle řešeno pomocí cyklu a jednotlivého porovnávání
- co takhle použít *llListFindList*

Změna parametrů uživatelem

- obvykle řešeno pomocí *llListen* na kanálu 1
- co takhle použít *llDialog* a *timer*

Další zdroje o LSL skriptování

Informace pro optimalizaci skriptů

- LSL chybová hlášení
- Paměťová náročnost kódu
- Rychlost běhu skriptu

Programování LSL skriptů

- Alternativní LSL editory
- Videonávody, psané návody, inworld návody
- Second Life tutorials

Další zdroje o LSL skriptování

Příklady a knihovny

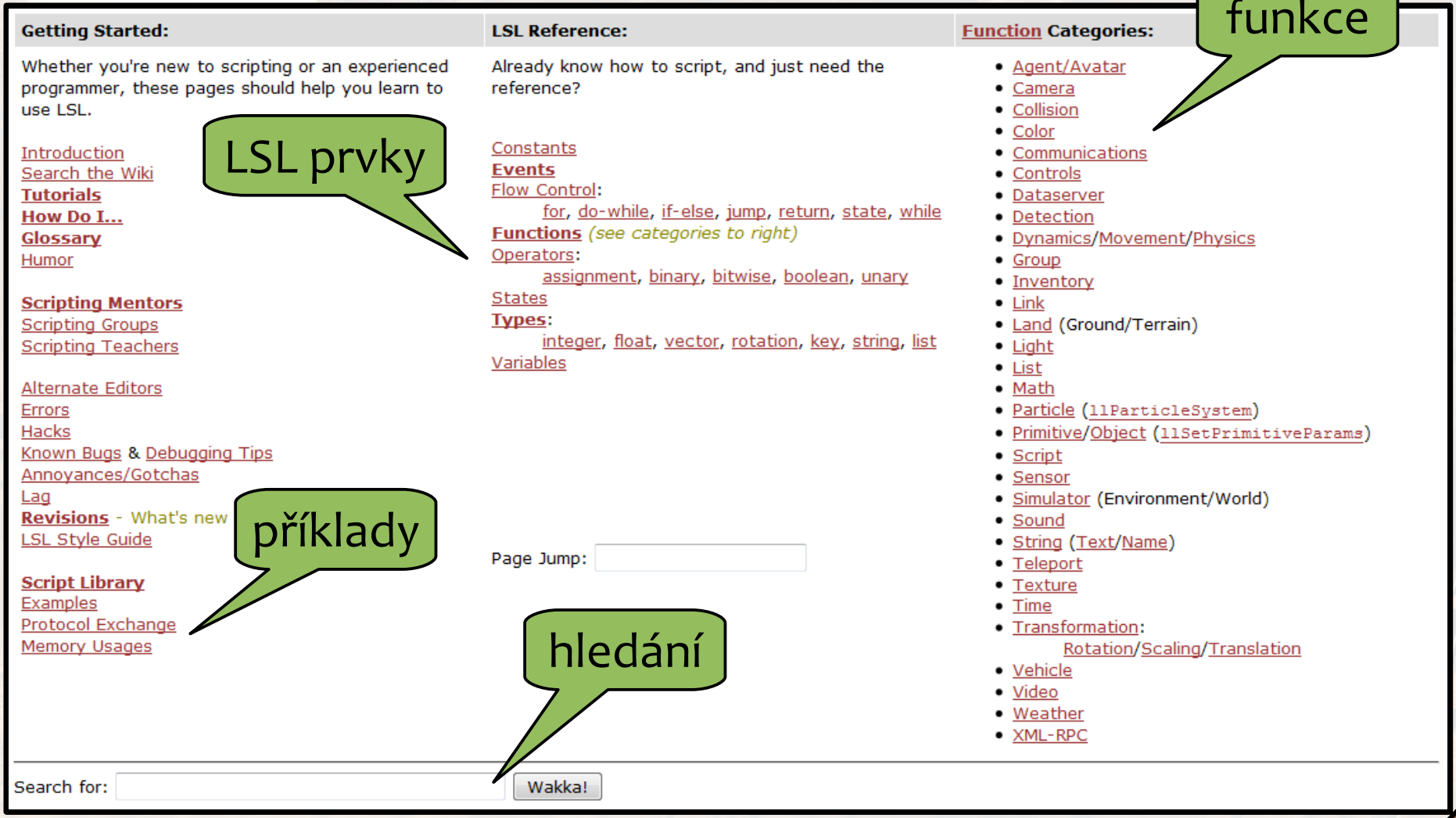
- Official Script Library
- SimTeach library
- Free SL scripts
- 2lifeblog Scripts
- ArianeB Basic Free Scripts
- Xstreet SL scripts
- Mitsi Script Library

LSL wiki

- obsahuje strukturovaně uspořádaný seznam předdefinovaných LSL funkcí včetně povinných vstupních parametrů a výstupních hodnot (a oproti oficiální LL wiki i velkou spoustu vysvětlujících textů, příkladů a poznámek)

- Link: http://www.lslwiki.net/lslwiki/

LSL wiki


Getting Started:
Whether you're new to scripting or an experienced programmer, these pages should help you learn to use LSL.
Introduction
Search the Wiki
Tutorials
How Do I...
Glossary
Humor
Scripting Mentors
Scripting Groups
Scripting Teachers
Alternate Editors
Errors
Hacks
Known Bugs & Debugging Tips
Annoyances/Gotchas
Lag
Revisions - What's new
LSL Style Guide
Script Library
Examples
Protocol Exchange
Memory Usages
LSL Reference:
Already know how to script, and just need the reference?
Constants
Events
Flow Control:
for, do-while, if-else, jump, return, state, while
Functions (see categories to right)
Operators:
assignment, binary, bitwise, boolean, unary
States
Types:
integer, float, vector, rotation, key, string, list
Variables
Page Jump:
Function Categories:
Agent/Avatar
Camera
Collision
Color
Communications
Controls
Dataserver
Detection
Dynamics/Movement/Physics
Group
Inventory
Link
Land (Ground/Terrain)
Light
List
Math
Particle (llParticleSystem)
Primitive/Object (llSetPrimitiveParams)
Script
Sensor
Simulator (Environment/World)
Sound
String (Text/Name)
Teleport
Texture
Time
Transformation:
Rotation/Scaling/Translation
Vehicle
Video
Weather
XML-RPC
Search for:
Wakka!
LSL prvky
příklady
hledání
funkce

Otázky a diskuze