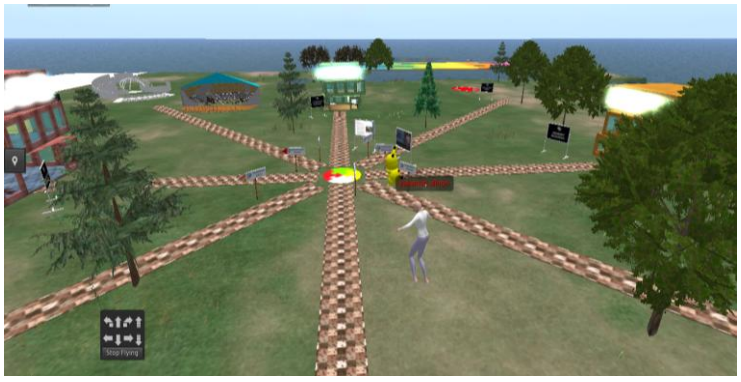
Middle East Technical University
Graduate School of Natural and Applied Sciences
Department of Computer Education and Instructional Technology

CEIT609 Virtual Worlds in Education Theory and Design

*"**Scripts**" for "**Brain Training Center**"*



İrem Gökçe AYDIN
Rukiye ALTIN
Tuğba ALTAN
Yasaman ALIOON

2013, Ankara

Table of Contents

SCRIPTS FOR BRAIN TRANING CENTER

1. NoteCard Giver

Figure 1- Notecard Giver example

After creating a notecard in the inventory under the “Notecards” section by right clicking and selecting “New Notecard”, a notecard content is filled. After the following script is attached to the object. The called notecard should be attached to the object so that from the script its name can be called.

```
{
    touch_start(integer total_number)
    {
        llGiveInventory(llDetectedKey(0), "Greeting");
    }
}
```

2. SlideShows

Figure 2 - Presentation Example

A power-point presentation slides can be published as images. To do this, in PowerPoint, click save as buton and select “Other Formats”, click one of image file format you want. After drag-drop slide media pictures in .jpg, .png formats under the Texture folder in the inventory, you should copy them under the presentation object’s content tab. After naming them in a meaningful manner and linking the slideshow object parts in order, attach the following script to your object:

```
integer pCurrentSlide = 0;
integer pSlideCount;
key pLastId;
integer dialog_channel= 1751;

default
{
    state_entry()
    {
         llListen(dialog_channel,"", "","");
         integer number = llGetInventoryNumber(INVENTORY_TEXTURE);
         pSlideCount = number;
```

```
         string      name      =      llGetInventoryName(INVENTORY_TEXTURE,
pCurrentSlide);
         llSetTexture(name, 3);
    }
    touch_start(integer total_number)
    {
        pLastId = llDetectedKey(0);
        integer number = llGetInventoryNumber(INVENTORY_TEXTURE);
        pSlideCount = number;
        string button = llGetLinkName(llDetectedLinkNumber(0));

        if (button == "back")
        {
            if (pCurrentSlide > 0)
            {
                pCurrentSlide = pCurrentSlide - 1;
            }else{
                pCurrentSlide = pSlideCount - 1;
            }
        }
        if (button == "open")
        {
            pCurrentSlide = 0;
        }
        if (button == "next")
        {
            if (pCurrentSlide < pSlideCount)
            {
                pCurrentSlide += 1;
            }else{
                pCurrentSlide = 0;
            }
        }
        string name = llGetInventoryName(INVENTORY_TEXTURE, pCurrentSlide);
        if (name != "") {
            llSetTexture(name, 3);
        }else{
            pCurrentSlide = 0;
            name = llGetInventoryName(INVENTORY_TEXTURE,pCurrentSlide);
            llSetTexture(name, 3);
        }
    }
}
```

This script gets the images from the inventory texture of the object and according to the count of them, a counter is increased-decreased according to the pressed linked button. For more refereces:

http://fleep.wikispaces.com/PPT+Slideshow+Opensim

3. Teleporter

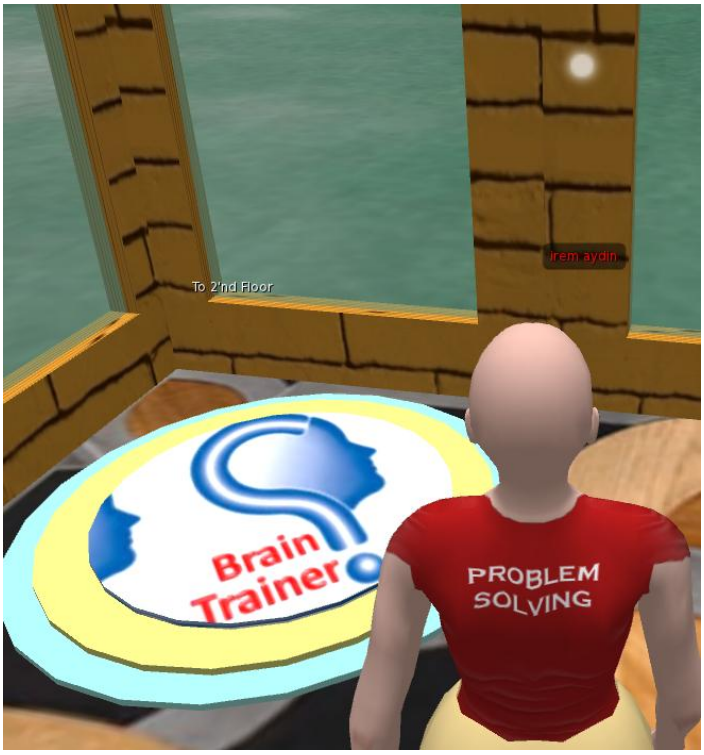


Figure 3 - Teleporter Example

For teleporting from somepoint, by clicking the object and selecting teleport option can be enabled with the following script:

```
vector targetPos = <174, 130, 26>; //The target teleport location
integer PERMISSION_TELEPORT = 0x1000;
string fltText = "To 2'nd Floor"; //The string that will appear on the
object
key lastAVkey = NULL_KEY;
reset()
{
    llSetSitText("Teleport");  //Instead of sitting string when right
clicking on the object, it says Teleport anymore.
    llSetText(fltText, <1,1,1>, 1);
    vector target;
    target = (targetPos- llGetPos()) * (ZERO_ROTATION / llGetRot());
    llSitTarget(target, ZERO_ROTATION);
}

perform(){
    llUnSit(llAvatarOnSitTarget());
    reset();
}
default
{
    state_entry()
    {

        reset();
```

```
    }

    on_rez(integer startup_param)
    {
        reset();
    }

    changed(integer change)
    {
        perform();
    }
    touch_start(integer i)
    {
        llSay(0,"Right click me and chose 'Teleport'");
        key teleportee = llDetectedKey(0);
        llRequestPermissions(teleportee, PERMISSION_TELEPORT);
    }
    run_time_permissions(integer perm)
    {
        if(PERMISSION_TELEPORT & perm)
        {
            //perform();
            //osTeleportAgent(llDetectedKey(0),targetPos,ZERO_VECTOR);
            //llTeleportAgent(teleportee, "", <93,159,23>, <93,159,23>);
        }
    }
}
```

This script uses llSitTarget function instead of teleportation. So after teleportation, unSit function is called. Because when we try the osTeleportAgent or llTeleportAgent functions, they did not work.

4. Game#1 – Memory Card (in Memory Building)

Figure 4 - Memory Card Game

In this game classical card matching implementation is applied. For now 3*3 colored cards are used. After linking the buttons&cards&game console object, main controller script of the game is assigned the overall object. When clicking the menu buton a Dialog Menu appears. Here is the script for Memory Card:

```
key pLastId;
integer dialog_channel= 1751;

// for menu
list dialogButtons = ["How To Play", "Credits"];
string message = "\nPlease make a choice.";   // The newline (\n) helps to
visually separate this text from the dialog heading line
key ToucherID;
integer channelDialog;
integer listenHandle;
// for menu
integer timeCounterShowCards = 3;
integer timeCounterGame = 0;
integer openCardNumber = 0;
integer matchedCardPairNumber = 0;
integer gameStarted = FALSE;

integer card1_state = FALSE; //close
integer card1_1_state = FALSE; //close
integer card2_state = FALSE; //close
integer card2_2_state = FALSE; //close


showCards(){
    llSetLinkAlpha(5, 1.0, ALL_SIDES);//cards
    llSetLinkAlpha(6, 1.0, ALL_SIDES);//cards
    llSetLinkAlpha(7, 1.0, ALL_SIDES);//cards
    llSetLinkAlpha(8, 1.0, ALL_SIDES);//cards
}

hideCards(){
    llSetLinkAlpha(5, 0.0, ALL_SIDES);//cards
    llSetLinkAlpha(6, 0.0, ALL_SIDES);//cards
    llSetLinkAlpha(7, 0.0, ALL_SIDES);//cards
    llSetLinkAlpha(8, 0.0, ALL_SIDES);//cards

}

closeCards(){
    llSetLinkPrimitiveParams( 5, [PRIM_ROT_LOCAL, llEuler2Rot(<0, 0, 180> *
DEG_TO_RAD)     *     llList2Rot(     llGetLinkPrimitiveParams(     LINK_ROOT,
[PRIM_ROT_LOCAL] ), 0 )] )    ;
    llSetLinkPrimitiveParams( 6, [PRIM_ROT_LOCAL, llEuler2Rot(<0, 0, 180> *
DEG_TO_RAD)     *     llList2Rot(     llGetLinkPrimitiveParams(     LINK_ROOT,
[PRIM_ROT_LOCAL] ), 0 )] )    ;
    llSetLinkPrimitiveParams( 7, [PRIM_ROT_LOCAL, llEuler2Rot(<0, 0, 180> *
DEG_TO_RAD)     *     llList2Rot(     llGetLinkPrimitiveParams(     LINK_ROOT,
[PRIM_ROT_LOCAL] ), 0 )] )    ;
    llSetLinkPrimitiveParams( 8, [PRIM_ROT_LOCAL, llEuler2Rot(<0, 0, 180> *
DEG_TO_RAD)     *     llList2Rot(     llGetLinkPrimitiveParams(     LINK_ROOT,
[PRIM_ROT_LOCAL] ), 0 )] )    ;
card1_state = FALSE;
card1_1_state = FALSE;
card2_state = FALSE;
```

```
card2_2_state = FALSE;
}

openCards(){
    llSetLinkPrimitiveParams( 5, [PRIM_ROT_LOCAL, llEuler2Rot(<0, 0, 0> *
DEG_TO_RAD)    *    llList2Rot(    llGetLinkPrimitiveParams(    LINK_ROOT,
[PRIM_ROT_LOCAL] ), 0 )] )   ;
    llSetLinkPrimitiveParams( 6, [PRIM_ROT_LOCAL, llEuler2Rot(<0, 0, 0> *
DEG_TO_RAD)    *    llList2Rot(    llGetLinkPrimitiveParams(    LINK_ROOT,
[PRIM_ROT_LOCAL] ), 0 )] )   ;
    llSetLinkPrimitiveParams( 7, [PRIM_ROT_LOCAL, llEuler2Rot(<0, 0, 0> *
DEG_TO_RAD)    *    llList2Rot(    llGetLinkPrimitiveParams(    LINK_ROOT,
[PRIM_ROT_LOCAL] ), 0 )] )   ;
    llSetLinkPrimitiveParams( 8, [PRIM_ROT_LOCAL, llEuler2Rot(<0, 0, 0> *
DEG_TO_RAD)    *    llList2Rot(    llGetLinkPrimitiveParams(    LINK_ROOT,
[PRIM_ROT_LOCAL] ), 0 )] )   ;
card1_state = TRUE;
card1_1_state = TRUE;
card2_state = TRUE;
card2_2_state = TRUE;
}

checkFinishCondition(){
    llSay(0,"matching card number " + (string)matchedCardPairNumber);
    if(matchedCardPairNumber == 2){
        llSetText("You have finished the game in "+(string)timeCounterGame
+ " seconds", <1,1,1>, 1);                    llResetScript();
        llSetTimerEvent(0);
    }
}

default
{
    state_entry()
    {
        llListen(dialog_channel,"", "","");
        integer card_number = llGetInventoryNumber(INVENTORY_TEXTURE);
        //llSay(0, "Blank Panel Link Number:" + (string)llGetLinkNumber());
        channelDialog   =   -1   -   (integer)("0x"   +   llGetSubString(
(string)llGetKey(), -7, -1) );
        llSetTimerEvent(0);
    }
    touch_start(integer total_number)
    {

        //llSay(0, (string)llDetectedLinkNumber(0));
        string button = llGetLinkName(llDetectedLinkNumber(0));
        list tButtonList = ["gameBoard","start","stop","menu"];
        list tTest = [(button)];
        integer foundIndex = llListFindList(tButtonList, tTest);
        //llSay(0, "Blank Panel Link Number:" + (string)foundIndex);
        //llSetLocalRot(llEuler2Rot(<0.0, 0.0, 60.0>));


        if (button == "start")
        {
            //llSay(0, "start clicked");
            llSetLinkAlpha(4, 0.0, ALL_SIDES);//menu button
            llSetLinkAlpha(2, 0.0, ALL_SIDES);//start button
            llSetLinkAlpha(3, 1.0, ALL_SIDES);//stop button
```

```
            showCards();
            llSetTimerEvent(0);
            llSetTimerEvent(1);
            llSetText("You have "+(string)timeCounterShowCards + " seconds
left", <1,1,1>, 1);

        }
        if (button == "menu")
        {
            //llSay(0, "menu clicked");
            ToucherID = llDetectedKey(0);
            listenHandle = llListen(channelDialog, "", ToucherID, "");
            llDialog(ToucherID, message, dialogButtons, channelDialog);
        }
        if(button == "stop"){
            //llSay(0, "stop clicked");
            llSetLinkAlpha(4, 1.0, ALL_SIDES);//menu button
            llSetLinkAlpha(2, 1.0, ALL_SIDES);//start button
            llSetLinkAlpha(3, 0.0, ALL_SIDES);//stop button
            hideCards();
            openCards();
            llSetText("", <1,1,1>, 1);
            llSetTimerEvent(0.0);
            llResetScript();
        }
        if(button == "card1"){
            if(openCardNumber == 2){
                closeCards();
                openCardNumber = 0;
            }
            else if(!card1_state){
                llSetLinkPrimitiveParams(        5,        [PRIM_ROT_LOCAL,
llEuler2Rot(<0, 0, 0> * DEG_TO_RAD) * llList2Rot( llGetLinkPrimitiveParams(
LINK_ROOT, [PRIM_ROT_LOCAL] ), 0 )] )   ;
                card1_state = TRUE;
                openCardNumber++;
                if(card1_1_state){
                    llSetLinkAlpha(5, 0.0, ALL_SIDES);//start button
                    llSetLinkAlpha(8, 0.0, ALL_SIDES);//stop button
                    openCardNumber = 0;
                    matchedCardPairNumber++;
                    checkFinishCondition();
                }
            }
            else if(card1_state){
                llSetLinkPrimitiveParams(        5,        [PRIM_ROT_LOCAL,
llEuler2Rot(<0,     0,     180>     *     DEG_TO_RAD)     *     llList2Rot(
llGetLinkPrimitiveParams( LINK_ROOT, [PRIM_ROT_LOCAL] ), 0 )] )   ;
                card1_state = FALSE;
            }
        }
        if(button == "card1_1"){
            if(openCardNumber == 2){
                closeCards();
                openCardNumber = 0;
            }
            else if(!card1_1_state){
                llSetLinkPrimitiveParams(        8,        [PRIM_ROT_LOCAL,
llEuler2Rot(<0, 0, 0> * DEG_TO_RAD) * llList2Rot( llGetLinkPrimitiveParams(
LINK_ROOT, [PRIM_ROT_LOCAL] ), 0 )] )   ;
                card1_1_state = TRUE;
```

```
                openCardNumber++;
                if(card1_state){
                    llSetLinkAlpha(5, 0.0, ALL_SIDES);//start button
                    llSetLinkAlpha(8, 0.0, ALL_SIDES);//stop button
                    openCardNumber = 0;
                    matchedCardPairNumber++;
                    checkFinishCondition();
                }
            }
            else if(card1_1_state){
                llSetLinkPrimitiveParams(        8,        [PRIM_ROT_LOCAL,
llEuler2Rot(<0,     0,     180>     *     DEG_TO_RAD)     *     llList2Rot(
llGetLinkPrimitiveParams( LINK_ROOT, [PRIM_ROT_LOCAL] ), 0 )] )   ;
                card1_1_state = FALSE;
            }
        }
        if(button == "card2"){
            if(openCardNumber == 2){
                closeCards();
                openCardNumber = 0;
            }
            else if(!card2_state){
                llSetLinkPrimitiveParams(        6,        [PRIM_ROT_LOCAL,
llEuler2Rot(<0, 0, 0> * DEG_TO_RAD) * llList2Rot( llGetLinkPrimitiveParams(
LINK_ROOT, [PRIM_ROT_LOCAL] ), 0 )] )   ;
                card2_state = TRUE;
                openCardNumber++;
                if(card2_2_state){
                    llSetLinkAlpha(6, 0.0, ALL_SIDES);//start button
                    llSetLinkAlpha(7, 0.0, ALL_SIDES);//stop button
                    openCardNumber = 0;
                    matchedCardPairNumber++;
                    checkFinishCondition();
                }
            }
            else if(card2_state){
                llSetLinkPrimitiveParams(        6,        [PRIM_ROT_LOCAL,
llEuler2Rot(<0,     0,     180>     *     DEG_TO_RAD)     *     llList2Rot(
llGetLinkPrimitiveParams( LINK_ROOT, [PRIM_ROT_LOCAL] ), 0 )] )   ;
                card2_state = FALSE;
            }
        }
        if(button == "card2_2"){
            if(openCardNumber == 2){
                closeCards();
                openCardNumber = 0;
            }
            else if(!card2_2_state){
                llSetLinkPrimitiveParams(        7,        [PRIM_ROT_LOCAL,
llEuler2Rot(<0, 0, 0> * DEG_TO_RAD) * llList2Rot( llGetLinkPrimitiveParams(
LINK_ROOT, [PRIM_ROT_LOCAL] ), 0 )] )   ;
                card2_2_state = TRUE;
                openCardNumber++;
                if(card2_state){
                    llSetLinkAlpha(6, 0.0, ALL_SIDES);//start button
                    llSetLinkAlpha(7, 0.0, ALL_SIDES);//stop button
                    openCardNumber = 0;
                    matchedCardPairNumber++;
                    checkFinishCondition();
                }
            }
```

```
            else if(card2_2_state){
                llSetLinkPrimitiveParams(        7,        [PRIM_ROT_LOCAL,
llEuler2Rot(<0,     0,     180>     *     DEG_TO_RAD)     *     llList2Rot(
llGetLinkPrimitiveParams( LINK_ROOT, [PRIM_ROT_LOCAL] ), 0 )] )   ;
                card2_2_state = FALSE;
            }
        }
    }
     listen(integer channel, string name, key id, string message)
    {

        llListenRemove(listenHandle);

        if (message == "How To Play")
        {
            llSay(0,"Just match the cards according to their colors");
        }
        else if(message == "Credits"){
            llSay(0,"We are the Brain Training Team!");
        }
    }
    timer()
    {
        //llSay(0,"counting come " + (string)gameStarted);
        if(!gameStarted){
            timeCounterShowCards--;
            llSetText("You have "+(string)timeCounterShowCards + " seconds
left", <1,1,1>, 1);
            if(timeCounterShowCards == 0){
                //  stop timer
                llSetTimerEvent(0.0);

                llListenRemove(listenHandle);
                llWhisper(0, "cards are closed");
                llSetText("", <1,1,1>, 1);
                closeCards();
                gameStarted = TRUE;
                llSetTimerEvent(1); //for game time counting
            }
        }
        else{
            timeCounterGame++;
            llSetText("You have spent "+(string)timeCounterGame + " seconds
in game", <1,1,1>, 1);
        }
    }

}
```

5. Game#2 – Lost in Migration (In Attention Building)

Figure 5 - Lost in Migration Game

In this game, because it is in Attention Building, the birds that are oriented differently from all of them is tried to be realized. In the gamet he directions for answers are given in dialog box buttons. According to the correct answers, the score is announced by log string and text over the object. 6 predefined question textures are added to the object. According to the texture item number, the answers are known, so the results are determined with them. Here is the script fort his game:

```
key pLastId;
integer dialog_channel= 1751;

// for menu
list buttons_menu = ["How To Play", "Credits"];
string message_menu = "\nPlease make a choice.";
string message_question = "\nPlease answer the question";

list q1_menu = ["up", "down","left","right"];
list q2_menu = ["up", "down","left","right"];
list q3_menu = ["up", "down","left","right"];
list q4_menu = ["up", "down","left","right"];
list q5_menu = ["up", "down","left","right"];

key ToucherID;
integer channelDialog;
```

```
integer listenHandle;
// for menu

integer totalNumberOfQuestion = 5;
integer totalNumberOfSuccess = 0;
integer timeCounterGame = 0;

integer questionTexture = 0;

finishGame(){
    llSetText("You have answered " + (string)totalNumberOfSuccess +
"/"+(string)totalNumberOfQuestion+ " questions correctly.", <1,1,1>, 1);
    llResetScript();
}

default
{
    state_entry()
    {
        totalNumberOfQuestion = llGetInventoryNumber(INVENTORY_TEXTURE)-1;
        llSay(0, "Blank Panel Link Number:" + (string)llGetLinkNumber());
        channelDialog = -1 - (integer)("0x" + llGetSubString(
(string)llGetKey(), -7, -1) );
    }
    touch_start(integer total_number)
    {
        llSay(0, (string)llDetectedLinkNumber(0));
        string button = llGetLinkName(llDetectedLinkNumber(0));
        list tButtonList = ["question","menu","start","stop"];
        list tTest = [(button)];
        integer foundIndex = llListFindList(tButtonList, tTest);
        llSay(0, "Blank Panel Link Number:" + (string)foundIndex);
        //llSetLocalRot(llEuler2Rot(<0.0, 0.0, 60.0>));


        if (button == "start")
        {
            llSay(0, "start clicked");
            llSetLinkAlpha(4, 1.0, ALL_SIDES);//stop button
            llSetLinkAlpha(1, 0.7, ALL_SIDES);//question button
            llSetLinkAlpha(2, 0.0, ALL_SIDES);//menu button
            llSetLinkAlpha(3, 0.0, ALL_SIDES);//start button

            //llSetTimerEvent(1);
            string name = llGetInventoryName(INVENTORY_TEXTURE,
questionTexture);
            llSay(0,name);
            llSetTexture(name, 3);

        }
        if (button == "menu")
        {
            llSay(0, "menu clicked");
            ToucherID = llDetectedKey(0);
            listenHandle = llListen(channelDialog, "", ToucherID, "");
            llDialog(ToucherID, message_menu, buttons_menu, channelDialog);
        }
        if(button == "stop"){
            llSay(0, "stop clicked");
            llSetLinkAlpha(2, 1.0, ALL_SIDES);//menu button
            llSetLinkAlpha(3, 1.0, ALL_SIDES);//start button
```

```
            llSetLinkAlpha(4, 0.0, ALL_SIDES);//stop button;
            llSetLinkAlpha(1, 0.0, ALL_SIDES);//question button
            llSetText("", <1,1,1>, 1);
            llResetScript();
        }
        if(button == "question"){
            if(questionTexture == totalNumberOfQuestion)
                finishGame();
            else
                questionTexture++;

            string     name     =     llGetInventoryName(INVENTORY_TEXTURE,
questionTexture);
            llSay(0,name);
            llSetTexture(name, 3);

            ToucherID = llDetectedKey(0);
            listenHandle = llListen(channelDialog, "", ToucherID, "");
            if(questionTexture == 1)
                llDialog(ToucherID,        message_question,        q1_menu,
channelDialog);
            if(questionTexture == 2)
                llDialog(ToucherID,        message_question,        q2_menu,
channelDialog);
            if(questionTexture == 3)
                llDialog(ToucherID,        message_question,        q3_menu,
channelDialog);
            if(questionTexture == 4)
                llDialog(ToucherID,        message_question,        q4_menu,
channelDialog);
            if(questionTexture == 5)
                llDialog(ToucherID,        message_question,        q5_menu,
channelDialog);
        }

    }
     listen(integer channel, string name, key id, string message)
    {

        llListenRemove(listenHandle);
        if(questionTexture == 1){
            if (message == "up")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }
        if(questionTexture == 2){
            if (message == "left")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }if(questionTexture == 3){
            if (message == "right")
```

```
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }
        if(questionTexture == 4){
            if (message == "up")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }
        if(questionTexture == 5){
            if (message == "up")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }

    }
    timer()
    {
        timeCounterGame++;
        llSetText("You have spent "+(string)timeCounterGame + " seconds in
game", <1,1,1>, 1);
    }

}
```

6. Game#3 – Calculation Nation (In Problem Solving Building)

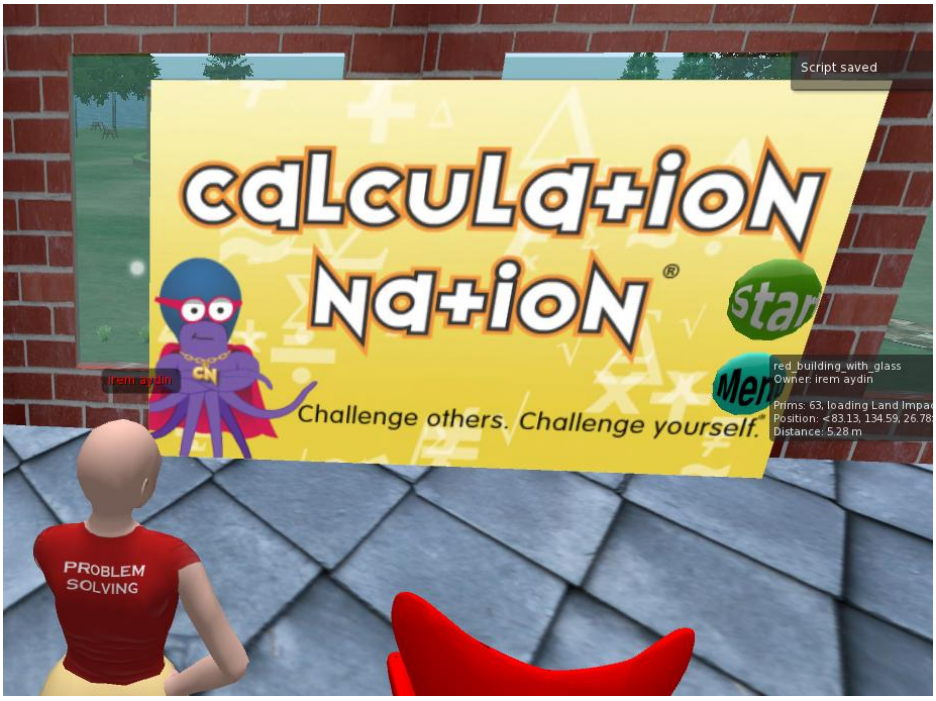


Figure 6 - Calculation Nation Game

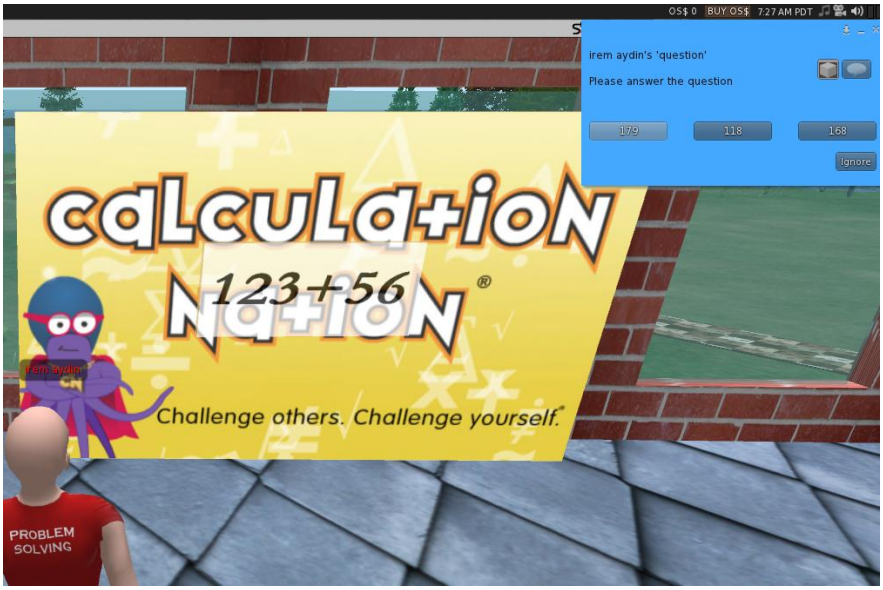
In this game some predefined 6 mathematical calculation problem is asked. The question textures are assigned to the game object as in Attention Building game. Here is the script of this game :

```
key pLastId;
integer dialog_channel= 1751;

// for menu
list buttons_menu = ["How To Play", "Credits"];
string message_menu = "\nPlease make a choice.";
string message_question = "\nPlease answer the question";

list q1_menu = ["75", "85","68"];
list q2_menu = ["179", "118","168"];
list q3_menu = ["337", "328","329"];
list q4_menu = ["16", "19","10"];
list q5_menu = ["1883", "1886","1786"];

key ToucherID;
integer channelDialog;
integer listenHandle;
// for menu

integer totalNumberOfQuestion = 5;
integer totalNumberOfSuccess = 0;
integer timeCounterGame = 0;

integer questionTexture = 0;
```

```
finishGame(){
    llSetText("You have answered " + (string)totalNumberOfSuccess +
"/"+(string)totalNumberOfQuestion+ " questions correctly.", <1,1,1>, 1);
    llResetScript();
}

default
{
    state_entry()
    {
        totalNumberOfQuestion = llGetInventoryNumber(INVENTORY_TEXTURE)-1;
        channelDialog = -1 - (integer)("0x" + llGetSubString(
(string)llGetKey(), -7, -1) );
    }
    touch_start(integer total_number)
    {
        llSay(0, (string)llDetectedLinkNumber(0));
        string button = llGetLinkName(llDetectedLinkNumber(0));
        list tButtonList = ["question","menu","start","stop"];
        list tTest = [(button)];
        integer foundIndex = llListFindList(tButtonList, tTest);
        //llSetLocalRot(llEuler2Rot(<0.0, 0.0, 60.0>));


        if (button == "start")
        {
            llSetLinkAlpha(4, 1.0, ALL_SIDES);//stop button
            llSetLinkAlpha(1, 0.7, ALL_SIDES);//question button
            llSetLinkAlpha(2, 0.0, ALL_SIDES);//menu button
            llSetLinkAlpha(3, 0.0, ALL_SIDES);//start button

            //llSetTimerEvent(1);
            string name = llGetInventoryName(INVENTORY_TEXTURE,
questionTexture);
            llSay(0,name);
            llSetTexture(name, 3);

        }
        if (button == "menu")
        {
            ToucherID = llDetectedKey(0);
            listenHandle = llListen(channelDialog, "", ToucherID, "");
            llDialog(ToucherID, message_menu, buttons_menu, channelDialog);
        }
        if(button == "stop"){
            llSetLinkAlpha(2, 1.0, ALL_SIDES);//menu button
            llSetLinkAlpha(3, 1.0, ALL_SIDES);//start button
            llSetLinkAlpha(4, 0.0, ALL_SIDES);//stop button;
            llSetLinkAlpha(1, 0.0, ALL_SIDES);//question button
            llSetText("", <1,1,1>, 1);
            llResetScript();
        }
        if(button == "question"){
            if(questionTexture == totalNumberOfQuestion)
                finishGame();
            else
                questionTexture++;

            string name = llGetInventoryName(INVENTORY_TEXTURE,
questionTexture);
            llSay(0,name);
```

```
            llSetTexture(name, 3);

            ToucherID = llDetectedKey(0);
            listenHandle = llListen(channelDialog, "", ToucherID, "");
            if(questionTexture == 1)
                llDialog(ToucherID,        message_question,        q1_menu,
channelDialog);
            if(questionTexture == 2)
                llDialog(ToucherID,        message_question,        q2_menu,
channelDialog);
            if(questionTexture == 3)
                llDialog(ToucherID,        message_question,        q3_menu,
channelDialog);
            if(questionTexture == 4)
                llDialog(ToucherID,        message_question,        q4_menu,
channelDialog);
            if(questionTexture == 5)
                llDialog(ToucherID,        message_question,        q5_menu,
channelDialog);
        }

    }
     listen(integer channel, string name, key id, string message)
    {

        llListenRemove(listenHandle);
        if(questionTexture == 1){
            if (message == "85")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }
        if(questionTexture == 2){
            if (message == "179")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }if(questionTexture == 3){
            if (message == "329")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }
        if(questionTexture == 4){
            if (message == "16")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
```

```
            else{
                llSay(0,"Wrong Answer!");
                }
            }
            if(questionTexture == 5){
                if (message == "1886")
                {
                    llSay(0,"True Answer!");
                    totalNumberOfSuccess++;
                }
                else{
                    llSay(0,"Wrong Answer!");
                }
            }

    }
    timer()
    {
        timeCounterGame++;
        llSetText("You have spent "+(string)timeCounterGame + " seconds in
game", <1,1,1>, 1);
    }

}
```



The answers for the questions are chosen by the dialog box buttons.

7. Game#4 – Calculation Nation (In Speed Building)

This game is the same one with in the Problem Solving Builging, but with time calculation. At the end of the questions, the correct answers are determined but with time restrictions. According to the time spend during answering the questions, at the end of the game, slow or speed conditions are expressed to the player by log string.

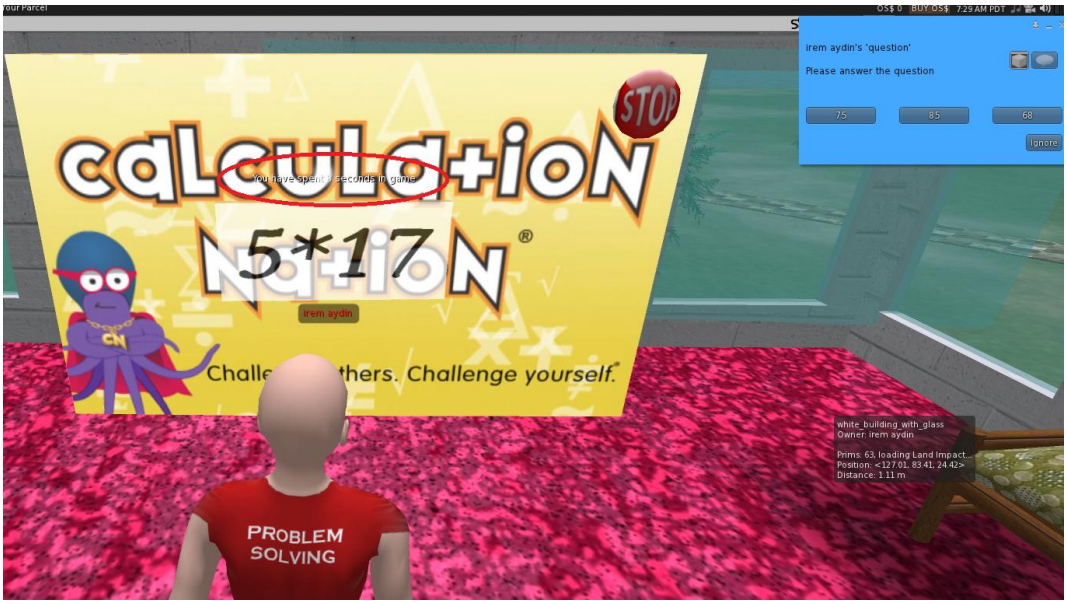


Figure 7 - Calculation Nation Game with Time

Here is the script for the game:

```
key pLastId;
integer dialog_channel= 1751;

// for menu
list buttons_menu = ["How To Play", "Credits"];
string message_menu = "\nPlease make a choice.";
string message_question = "\nPlease answer the question";

list q1_menu = ["75", "85","68"];
list q2_menu = ["179", "118","168"];
list q3_menu = ["337", "328","329"];
list q4_menu = ["16", "19","10"];
list q5_menu = ["1883", "1886","1786"];

key ToucherID;
integer channelDialog;
integer listenHandle;
// for menu

integer totalNumberOfQuestion = 5;
integer totalNumberOfSuccess = 0;
integer timeCounterGame = 0;

integer questionTexture = 0;

finishGame(){
    if(timeCounterGame < 5)
        llSay(0, "You are very fast!!!!");
    else if(timeCounterGame > 20)
        llSay(0, "You are very slow!!!!");
    else
        llSay(0, "Your speed is moderate..");
```

```
    llSetText("You have answered " + (string)totalNumberOfSuccess +
"/"+(string)totalNumberOfQuestion+ " questions correctly in "+
(string)timeCounterGame + " seconds.", <1,1,1>, 1);
    llResetScript();
}

default
{
    state_entry()
    {
        totalNumberOfQuestion = llGetInventoryNumber(INVENTORY_TEXTURE)-1;
        //llSay(0, "Blank Panel Link Number:" + (string)llGetLinkNumber());
        channelDialog = -1 - (integer)("0x" + llGetSubString(
(string)llGetKey(), -7, -1) );
    }
    touch_start(integer total_number)
    {
        //llSay(0, (string)llDetectedLinkNumber(0));
        string button = llGetLinkName(llDetectedLinkNumber(0));
        list tButtonList = ["question","menu","start","stop"];
        list tTest = [(button)];
        integer foundIndex = llListFindList(tButtonList, tTest);
        //llSay(0, "Blank Panel Link Number:" + (string)foundIndex);
        //llSetLocalRot(llEuler2Rot(<0.0, 0.0, 60.0>));


        if (button == "start")
        {
            //llSay(0, "start clicked");
            llSetLinkAlpha(4, 1.0, ALL_SIDES);//stop button
            llSetLinkAlpha(1, 0.7, ALL_SIDES);//question button
            llSetLinkAlpha(2, 0.0, ALL_SIDES);//menu button
            llSetLinkAlpha(3, 0.0, ALL_SIDES);//start button

            llSetTimerEvent(1);
            string name = llGetInventoryName(INVENTORY_TEXTURE,
questionTexture);
            //llSay(0,name);
            llSetTexture(name, 3);

        }
        if (button == "menu")
        {
            //llSay(0, "menu clicked");
            ToucherID = llDetectedKey(0);
            listenHandle = llListen(channelDialog, "", ToucherID, "");
            llDialog(ToucherID, message_menu, buttons_menu, channelDialog);
        }
        if(button == "stop"){
            //llSay(0, "stop clicked");
            llSetLinkAlpha(2, 1.0, ALL_SIDES);//menu button
            llSetLinkAlpha(3, 1.0, ALL_SIDES);//start button
            llSetLinkAlpha(4, 0.0, ALL_SIDES);//stop button;
            llSetLinkAlpha(1, 0.0, ALL_SIDES);//question button
            llSetText("", <1,1,1>, 1);
            llResetScript();
        }
        if(button == "question"){
            if(questionTexture == totalNumberOfQuestion)
                finishGame();
            else
```

```
                questionTexture++;

            string     name     =     llGetInventoryName(INVENTORY_TEXTURE,
questionTexture);
            //llSay(0,name);
            llSetTexture(name, 3);

            ToucherID = llDetectedKey(0);
            listenHandle = llListen(channelDialog, "", ToucherID, "");
            if(questionTexture == 1)
                llDialog(ToucherID,        message_question,        q1_menu,
channelDialog);
            if(questionTexture == 2)
                llDialog(ToucherID,        message_question,        q2_menu,
channelDialog);
            if(questionTexture == 3)
                llDialog(ToucherID,        message_question,        q3_menu,
channelDialog);
            if(questionTexture == 4)
                llDialog(ToucherID,        message_question,        q4_menu,
channelDialog);
            if(questionTexture == 5)
                llDialog(ToucherID,        message_question,        q5_menu,
channelDialog);
        }

    }
     listen(integer channel, string name, key id, string message)
    {

        llListenRemove(listenHandle);
        if(questionTexture == 1){
            if (message == "85")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }
        if(questionTexture == 2){
            if (message == "179")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }if(questionTexture == 3){
            if (message == "329")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }
        if(questionTexture == 4){
```

```
            if (message == "16")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }
        if(questionTexture == 5){
            if (message == "1886")
            {
                llSay(0,"True Answer!");
                totalNumberOfSuccess++;
            }
            else{
                llSay(0,"Wrong Answer!");
            }
        }

    }
    timer()
    {
        timeCounterGame++;
        llSetText("You have spent "+(string)timeCounterGame + " seconds in
game", <1,1,1>, 1);
    }

}
```

8. References

- http://fleep.wikispaces.com/Scripts
- http://opensimulator.org/wiki/Scripting_Documentation
- http://wiki.secondlife.com/wiki/Category:LSL_Library
- http://opensim-creations.com/category/scripts/
- http://www.freeslscripts.gendersquare.org/