```
#include "data/scripts/vars/index.h"    //Index variables.
#include "data/scripts/traileru.c"      //Shadow trails.
#include "data/scripts/com/ani0013.h"   //Jump animation if steping off an edge.

void main()
{
    void    vEnt;
      //Entity placeholder.
    void    vMap;
      //Color array placeholder.
    char    cName;
      //Entity default name.
    int     iAni;
      //Animations.
    int     iLiv    = -1;
      //Living enemies.
    int     iKMap;
      //KO map.
    int     iType;
      //Entity type.
    int     iVRes   = openborvariant("vresolution");
      //Current vertical resolution.
    int     iFrontZ = openborconstant("FRONTPANEL_Z");
      //Current front panel location.
    int     iECnt   = openborvariant("ent_max");
      //Current # of entities in play.
    int     iIndex;
      //Player index.
    int     iEnt;
      //Entity counter.
    int     iHSpr;
      //Sprite index.
    float   fCnt    = 0.0;
      //General counter.
    float   fJar;
      //Mp Jar count.
    float   health;
      //Current health
    float   fHPer;
      //HP % of max.
    float   fFron   = 0.0;
      //Front percentage (top 1/4 of HP)

    tupdate();

    for(iEnt=0; iEnt<iECnt; iEnt++)
      //Loop entity collection.
    {
        vEnt = getentity(iEnt);
      //Get entity handle.

        if(vEnt
      //Valid handle?
            && getentityproperty(vEnt, "exists")
      //Valid entity?
            && !getentityproperty(vEnt, "dead"))
      //Alive?
          {
            iType   = getentityproperty(vEnt, "type");
      //Get type.

            if(getentityproperty(vEnt, "owner")==NULL())
      //Not projectile?
            {
                if(getentityproperty(vEnt, "aiflag", "drop"))
      //Falling?
                {
```

```
                changeentityproperty(vEnt, "stealth", 1);
    //Set stealth.
            }
            else
            {
                cName   = getentityproperty(vEnt, "defaultname");
    //Get default name.
                iAni    = getentityproperty(vEnt, "animationid");
    //Get current animation.

                if(cName != "yamoto"
    //Not Yamoto?
                    &&  !(cName == "Alex" && iAni == openborconstant("ANI_RISEATTACK"
)) //Not Alex doing a rise attack?
                    &&  iAni != openborconstant("ANI_SLEEP"))
    //Not in sleep ani?
                {
                    changeentityproperty(vEnt, "stealth", 0);
    //Turn stealth off.
                }

                ani0013(vEnt, iAni, 0);
    //Auto jump from platforms.
            }
        }

        if ((iType && iType == openborconstant("TYPE_PLAYER")))
    //Player type?
        {
            iIndex  = getentityproperty(vEnt, "playerindex");
    //Get player index.
            fJar    = getentityproperty(vEnt, "mp")/10;
    //MP jar count.
            fHPer   = 4 * (0.0 + (hlife(vEnt)));
    //Get life % in quarters.
            iHSpr   = getindexedvar(ICOJAR);
    //Get magic jar sprite.

            for(fCnt=0; fCnt<fJar; fCnt++)
    //Loop jar count.
            {
                drawsprite(iHSpr, iIndex*160+55+fCnt*11, iVRes-20, iFrontZ+18001);
    //Draw magic jars
            }

            for(fCnt=0.0; fCnt<fHPer; fCnt++)
    //Loop each quater of life.
            {
                fFron   = fHPer - fCnt;
                iHSpr   = getindexedvar(lblock(fFron));
    //Get life block sprite.

                drawsprite(iHSpr, iIndex*160+53+fCnt*26, iVRes-31, iFrontZ+18001);
    //Draw life block.
            }
        }
        else
        {
            iHSpr   = getentityproperty(vEnt, "icon", 3);
    //Get icon sprite.

            if(iHSpr != -1)
    //Sprite valid?
            {
                fHPer   = hlife(vEnt);
    //Get life block sprite.
                vMap    = getentityproperty(vEnt, "colourmap");
```

```
                    ++iLiv;
      //Increment "living" index.

                    setdrawmethod(NULL(), 1, 256, 256, 0, 0, 0, 0, 0, 0, 0, 0, 0, vMap);
      //Set global draw method.
                    drawsprite(iHSpr, (iLiv*41), 2, iFrontZ+18000);
      //Draw icon.
                    setdrawmethod(NULL(), 0, 256, 256, 0, 0, 0, 0, 0, 0, 0, 0, 0, NULL());
      //Restore global draw defaults.

                    iHSpr   = getindexedvar(lblock(fHPer));
      //Get life block sprite.
                    drawsprite(iHSpr, 16+(iLiv*41), 8, iFrontZ+18000);
     //Draw life block.
                }
            }
        }
    }
}

float hlife(void vEnt)
{
    float fHP   = 0.0 + getentityproperty(vEnt, "health");
    float fMHP  = 0.0 + getentityproperty(vEnt, "maxhealth");

    return fHP/fMHP;
}

int lblock(float fPer){

    int iHSpr;

    if (fPer >= 0.75)
    {
        iHSpr = BLOCBLU;  //Blue
    }
    else if (fPer >= 0.50)
    {
        iHSpr = BLOCYEL;  //Yellow
    }
    else if (fPer >= 0.25)
    {
        iHSpr = BLOCORA;  //Orange
    }
    else
    {
        iHSpr = BLOCRED;  //Red
    }

    return iHSpr;
}
```