About Silver's RPG Making Guide v1.0.2

Original release: 19 October 07
Update release: 22 Apr 08

Thank you for downloading my guide on RPG making. If you have any comments or inquiries regarding it or if you wish to see more of my work you can contact me at: http://www.roguesoft.co.uk

You can download Mac GameMaker at: http://members.aol.com/AlStaff/GameMakerHelp.html

Blue text = Code
Red text = User Input

Intro To Game Designing

Welcome, and congratulations on getting this far! Most people who develop an interest in designing their own RPG give up before they even begin! There are some who might get as far as to learn a bit of programming, but they soon abandon all their hopes and dreams of moulding their very own dungeon exploring, treasure finding, monster slaying, EXP earning, level raising RPG because they get frustrated at not being able to program any of the things they want their game to have. It's a sad fact that this problem is usually caused by people being impatient, and quite often lazy.

The golden rule to remember is: start small. The reason why people can't program their super cool 100mb RPG right away on day one is simply that they don't know enough about programming. Get experience by making a few small games before you attempt the next World of Warcraft, and get used to the application you're using to create them with. (in this case Mac GameMaker, and it couldn't be a better choice)

I myself rushed into RPG designing the moment I got my hands on my first game editor years and years ago. I didn't understand anything about programming so the game was a complete flop! However I continued searching for and downloading game

editors, hoping that one of them would do for what I wanted. Little did I know that any one of them would have done for what I wanted. The problem was that I was trying to find a magic editor that would put everything I wanted into the game for me at the click of a few buttons, when what I *should* have been doing was learning programming.

Programming isn't that hard to learn at all, in fact provided you've got a good language (like Mac GameMaker's) it's quite simple. Once you know the rules that apply to it you can make pretty much any kind of game you like! This guide will help you get started creating your own RPG, but before you continue I advise you to spend a few days or a week getting used to Mac GameMaker's programming language. Make a few small games, get to know a few of the commands and variables, and what a variable *is*, hehe. Don't worry, I'll wait for you...

Tutorial On Mac GameMaker's Language

Although I've just told you to spend a few days getting used to Mac GameMaker's language, I'm guessing that you'll be too impatient to do so. (sigh...) As I've already mentioned, impatience can lead to frustration, and frustration can lead to giving up. To help prevent this, here's a quick rundown on how the language works. (anyone who knows how to use variables can skip this section)

Variables

The whole language centers around variables. A variable is a letter or a word that has an assigned value. To assign a variable the value of a *number*, you would type the following:

gold = 50

That would make the variable "gold" equal "50". To assign a variable the value of a *word*, you would type the following:

name$ = "Silverwind"

That would make the variable "name$" equal "Silverwind". Notice the $ sign at the end of the variable's name? That's required when assigning a variable the value of a *word* but not a number.

Printing Values

When you want to print the value of a variable, type the variable you want to print and incase it with $ marks. Like this:

PRINT $gold$

Because I've assigned "gold" a value of "50" earlier, the above line of code will print "50".

PRINT $name$$

Because I've assigned "name$" the value of "Silverwind" earlier, the above line of code will print "Silverwind". The same method is used to print values in *any* print form. These lines work as well:

ALERT You have $gold$ Gold.

POPFIELD My name is $name$$

Well, that's about it for variables. Simple eh? But before we go on, I still strongly advise you to spend a few days making small non-RPG games if you haven't done so already. Whenever you're up for it, move on to the next section.

Getting Started

Ok, so now that you now what variables are and how to use them, we can start making an RPG! To start off you'll need to have a vague idea of what you want to include in your game, and the trick is: start small. You can always add to it later on, but there's no point running a marathon before you learn to walk, meaning it's best to keep the overall goal for your first RPG simple. I recommend starting with the battle engine, as it's the foundry of most RPGs.

Your First Battle Engine

The battle engine is a great place to start. Let's say that the goal for your first battle engine is to have a player character and an enemy, each with their own HP, (Hit Points) and an Attack button that makes them attack each other. Ok, so create a new game in Mac GameMaker (GM as we'll now refer to it) and select a Card layout that includes a text field. In the button script of the Begin Game button on Card 1, create variables for both the player and the enemy's HP. Like this:

```
playerHP = 30
enemyHP = 25
```

Now go to Card 2 and enter this in the Card script:

```
CLEAR TEXT
PRINT Player HP: $playerHP$
PRINT Enemy HP: $enemyHP$
```

(the "CLEAR TEXT" line isn't necessary because the text field will usually be blank when the card loads, but I add it anyway incase I leave a note to myself in the text field in the editor) So now when Card 2 loads the screen should read "PlayerHP: 30 EnemyHP: 25". What's that I hear you say? "Cool"? You're right, it is cool, but what good are HP if you can't do anything with them? Let's create an Attack button so that the player and the enemy can damage each other! Create a button and call it "Attack" or "Fight" or whatever you want, and put the following code in it:

```
damage = RANDOM 5
enemyHP = enemyHP - damage
```

That will make the player attack the enemy, so lets add a bit more code and make the enemy attack the player afterwards.

```
damage = RANDOM 5
playerHP = playerHP - damage
```

Now that damage has been done to the HP of both the player and the enemy, lets tell the computer to print the updated values of their HP.

```
CLEAR TEXT
PRINT Player HP: $playerHP$
PRINT Enemy HP: $enemyHP$
```

Coolaboola! Now when you click the Attack button the player and the enemy damage each other and the new HP are printed! Awesome!!! But wait... uh oh, look at what happens if you click the Attack button over and over again: the HP drop to a value below 0! Not a problem, we'll add a few lines of code to tell the computer that if the HP of a character reaches 0 we want that character to die! (Nothing personal, but we can't have a load of "less then 0 HP" people running around in our game) Ok, here's what to do. Under the line that says "enemyHP = enemyHP - damage" type:

```
IF enemyHP < 1 THEN
  ALERT You have defeated the enemy! Hurray!
  GOTOCARD 1
END IF
```

Now make a similar block of code and put it under the line that says "playerHP = playerHP - damage"

```
IF playerHP < 1 THEN
  ALERT You have been defeated...
  GOTOCARD 1
END IF
```

Take a moment to step back and gaze at your creation... gasp! You have achieved your goal! You have made a player character and an enemy BOTH with HP, and you can make them damage each other by clicking the Attack button! Congratulations, you've programmed a simple battle engine! By changing the values of "damage", "playerHP" and "enemyHP" you can make the player and enemy's attack stronger or weaker. Cooooooool! When you're ready to build a slightly more flexible battle engine, move on to the next section.

Basic Battle Engine

Now that you've programmed your first battle engine, lets try building one that's slightly more flexible. You'll probably notice the code to be much the same as the last, but this one will use a minimum of 3 Cards and will be easier to import new features into. Ok, so just like with our first battle engine the player's stats have to be set *before* they enter a battle. (the Begin Game button on Card 1 is probably the best place to set

them) The following Cards will be used:

- Player's Turn Card.
- Enemy's Turn Card.
- Game Over Card.

You can work on any Cards you like, but for this tutorial we'll say that we're working on Cards 2 and 3. Enter this in the Card script on Card 2. (the Player's Turn Card)

```
CLEAR TEXT
PRINT stats
```

Now create an Attack button and enter this in it's script:

```
playerhit = RANDOM damage amount
enemyHP = enemyHP - playerhit

CLEAR TEXT
PRINT stats

IF enemyHP < 1 THEN
  ALERT You have defeated the enemy! Hurray!
  GOTOCARD Victory Card
ELSE
  GOTOCARD Enemy's Turn Card
END IF
```

Now go to Card 3 (the Enemy's Turn Card) and enter this in the Card script:

```
enemyhit = RANDOM damage amount
playerHP = playerHP - enemyhit

CLEAR TEXT
PRINT stats

IF playerHP < 1 THEN
  ALERT You have been defeated...
  GOTOCARD Game Over Card
ELSE
```

```
  GOTOCARD Player's Turn Card
END IF
```

Notice how I've replaced "damage" with "playerhit" and "enemyhit"? This is so that you can reference the strength of the player's attack and the enemy's attack separately if you ever need to. (although it's unlikely that you ever will) You can change these variables to whatever you like. You might prefer "playerdamage" or "playerD". I personally always use "playerhit" though. Finally, notice how Card 4 is loaded when the player is defeated? That's because Card 4 will serve as our Game Over Card. You don't need much code on that Card, just draw a game over picture and make a button that goes back to Card 1 or quits the game.

Well, there you have it, a basic battle engine! You can add to it with some of the code snippets from the RPG Code Resources section found at the back of this guide, or modify it with your own code.

More Than Just Code

Ok, once you've got the hand of programming you'll need to sit yourself down and consider the other aspects of your RPG, and one of the most important things to consider is the quest/story line. Having a plot or event take place at the *beginning* of an RPG is the most common way of setting the scene for the player. The great thing about the story line is that it's entirely up to you. You can take pretty much any random idea that comes into your head and build a game around it! Many people create really big story lines for their RPG, with dozens of characters and "complications" to detail it. But there are others (like myself) who prefer to go with a simple theme. Here's examples of the two most common types of story line:

Character Centered

Many story lines center around a single character, who's allegiance plays an important role in the goal of the game. For example, if the character's a good guy it might be a princess who gets kidnapped, the goal of the game being to rescue her. Or if the character's a bad guy, it might be a corrupt king or an evil wizard, the goal of the game being to overthrow his rule on the realm.

Item Centered

Another common theme is to center the story line around an item or artifact of immense power. (the Sword of the Legendary Crusader, the Amulet of Destruction, the Ring of Darkness etc.) As with character centered story lines, the item's alignment plays an important role on the game's goal. If the item serves an evil purpose, the player's goal might be to find and destroy the item before the enemy can use it. On the other hand, if the item serves a good purpose the player's goal might be to find and use it *against* the enemy. In this theme the item is often broken into several pieces or shards, meaning that the player has to journey all over the realm locating and retrieving the pieces before combining them and using the restored item to overthrow the central villain.

There are many more story line themes, but the two above are the most common. Here's an example story line that uses elements of both themes: "The evil wizard Zelderof has risen in power once more and is taking over the land with his army of undead minions! Only *you* can bring an end to his plans by retrieving all 7 pieces of the legendary Amulet of Light and ridding the world of the wizard's unholy existence!"

RPG Code Resources:

Most of the following codes require the Basic Battle Engine to execute correctly. I've listed them in alphabetical order but you can add as many or as little of them to your RPG as you want without any specific order. (with the exception of certain codes which require the combined use of other codes to execute correctly)

Escape Battle Button

Requirements:
• Basic Battle engine.
• "battlereturn" variable. (see Random Enemy Encounters)

Create an Escape or Run Away button on the Player's Turn Card in the battle engine and enter this in the button script:

escapechance = RANDOM 100

```
IF escapechance > chance of escaping THEN
  ALERT You escape the enemy...
  GOTOCARD
ELSE
  ALERT You can't escape!
  GOTOCARD battlereturn
END IF
```

EXP and LV's (experience points and player levels)

Requirements:
• Basic Battle Engine
• Random Enemy Encounters

Create an EXP Award Card that's accessed whenever an enemy is defeated in battle and enter this in the Card script:

```
gold = gold + enemydropgold
IF gold > 30000 THEN gold = 30000
playerEXP = playerEXP + enemyEXPvalue
IF playerEXP > 30000 THEN playerEXP = 30000
ALERT You have defeated the enemy and gained $enemyEXPvalue$ EXP and
$enemydropgold$ gold!

IF playerEXP > level 2 requirement THEN newLV = 2
IF playerEXP > level 3 requirement THEN newLV = 3
IF playerEXP > level 4 requirement THEN newLV = 4
IF playerEXP > level 5 requirement THEN newLV = 5

IF newLV > playerLV THEN
  HPup = RANDOM HP increase amount
  maxHP = maxHP + HPup
  IF maxHP > 30000 THEN maxHP = 30000
  playerHP = playerHP + HPup
  IF playerHP > maxHP THEN playerHP = maxHP
  ALERT You have reached level $newLV$! Your HP increased by $HPup$.
  playerLV = newLV
```

END IF

GOTOCARD battlereturn

Now edit the enemy information blocks on the Enemy Encounter Card to contain the following lines:

enemyEXPvalue = enemy's EXP value
enemydropgold = enemy's carried gold

Healing Potions

Requirements:
• *None*.

Create a Use Heal Potion button and enter this in the button script:

IF healpotions > 0 THEN
  healpotions = healpotions - 1
  healamount = RANDOM heal amount
  healamount = healamount + heal plus
  playerHP = playerHP + healamount
  IF playerHP > maxHP THEN playerHP = maxHP
  ALERT You've been healed $healamount$ HP!
ELSE
  ALERT You have no Heal Potions.
END IF

Items

Requirements:
• *None*.

Create an Items Card with buttons representing the names of each item the player can use, then enter this in each button script:

```
IF item name > 0 THEN
  item name = item name - 1
  item effect
  ALERT You've used a item name
ELSE
  ALERT You have no item name
END IF
```

Items (in battle)

Requirements:
• Basic Battle Engine
• "battlereturn" variable. (see Random Enemy Encounters)
• Items

To have items usable in battle, create a Back button on the Items Card and have it link to the battle Card. Now Create a button on your Battle Card called Use Item and have it link to the Items Card.

Player Attributes

Requirements:
• *None*.

Create an Attribute Distribution Card with buttons representing the names of each attribute in the game, then enter this in each attribute's button script:

```
IF totalpoints > 0 THEN
  attribute = attribute + 1
  totalpoints = totalpoints - 1
ELSE
  ALERT You have used up all of your attribute points!
END IF
```

CLEAR
PRINT Remaining Attribute Points: $totalpoints$
PRINT
PRINT attribute 1: $attribute 1$
PRINT attribute 2: $attribute 2$
PRINT attribute 3: $attribute 3$

Enter this in the Card script:

totalpoints = total amount of attribute points
attribute 1 = 0
attribute 2 = 0
attribute 3 = 0

CLEAR
PRINT Remaining Attribute Points: $totalpoints$
PRINT
PRINT attribute 1: $attribute 1$
PRINT attribute 2: $attribute 2$
PRINT attribute 3: $attribute 3$

Create a Reset button and enter this in it's script:

totalpoints = total amount of attribute points
attribute 1 = 0
attribute 2 = 0
attribute 3 = 0

CLEAR
PRINT Remaining Attribute Points: $totalpoints$
PRINT
PRINT attribute 1: $attribute 1$
PRINT attribute 2: $attribute 2$
PRINT attribute 3: $attribute 3$

Now create a Finished button and enter this in it's script:

IF totalpoints = 0 THEN
  ALERT You have finished assigning all of your attribute points.

  GOTOCARD game start Card
ELSE
  ALERT You haven't finished assigning all of your attribute points.
END IF

Player Classes

Requirements:
• *None.*

Create a Choose Player Class Card with buttons representing the names of each class the player can select, then enter this in each button script:

playerclass$ = "class name"

When you want to have a button's code execute only when a player is of a specific class, enter this in the button script:

IF playerclass$ = "class name" THEN
  class limited code
ELSE
  ALERT You need to be a class name to do this.
END IF

Here's an example class limited event: (placed in the Cast Spell button in the battle engine)

IF playerclass$ = "Wizard" THEN
  GOTOCARD Spell Select Card
ELSE
  ALERT You are not a Wizard! Only Wizards can cast spells!
END IF

Player Status

Requirements:
• *None.*

Create a variable named "playerstatus$" and appropriately place this code into your game: (placement varies on the nature of the status effect)

```
IF playerstatus$ = "status" THEN
  status effect
END IF
```

Here's an example status: (this code is placed at the beginning of the Attack button in the battle engine)

```
IF playerstatus$ = "Asleep" THEN
  awakechance = RANDOM 5
  IF awakechance = 1 THEN
    playerstatus$ = "Normal"
    ALERT You wake up!
  ELSE
    ALERT You are fast asleep...
    GOTOCARD Enemy's Turn Card
  END IF
END IF
```

Random Enemy Encounters

Requirements:
• Basic Battle Engine

Create an Encounter Card and enter this in the Card script:

```
battlereturn = RECENTCARD
enemyselect = RANDOM number of encounterable enemies

IF enemyselect = enemy number THEN
  enemy stats
END IF
```

Here's an example Encounter Card script: (the variables "enemyEXPvalue" and "enemydropgold" are for use with the "EXP and LV's" code)

```
battlereturn = RECENTCARD
enemyselect = RANDOM 3

IF enemyselect = 1 THEN
  enemyname$ = "Goblin"
  enemyHP = 10
  enemyEXPvalue = 5
  enemydropgold = RANDOM 5
END IF

IF enemyselect = 1 THEN
  enemyname$ = "Troll"
  enemyHP = 25
  enemyEXPvalue = 20
  enemydropgold = RANDOM 15
END IF

IF enemyselect = 1 THEN
  enemyname$ = "Dragon"
  enemyHP = 50
  enemyEXPvalue = 80
  enemydropgold = RANDOM 30
END IF
```

Enter this in the Card script of each Card you want to have a chance of encountering an enemy on:

```
encounterchance = RANDOM chance of encountering an enemy
IF encounterchance = 1 THEN GOTOCARD Encounter Card
```

Shops

Requirements:

• *None.*

Create a Shop Card with buttons representing the names of each item the player can buy, then enter this in each button script:

IF gold > item price - 1 THEN
  item = item + 1
  gold = gold - item price
  ALERT You purchase a item for item price gold.
ELSE
  ALERT You don't have enough gold!
END IF

Spells

Requirements:
• Basic Battle Engine
• "battlereturn" variable. (see Random Enemy Encounters)

Create a Spells Card with buttons representing the names of each spell the player can select, then enter this in each button script:

spell$ = "spell name"

Create a Back button on the Spells Card and have it link to the Battle Card. Create a Select Spell button on the Battle Card and have it link to the Spells Card. Now create a Cast Spell button on the Battle Card and enter this in it's script:

IF spell$ = "" THEN
  ALERT You don't have a spell selected!
END IF

IF spell$ = "spell name" THEN
    spell effect
END IF

If a spell has a HP damaging effect, put spelldamage = 1 at the end of it's effect.

Here's an example spell:

```
IF spell$ = "fire ball" THEN
  playerhit = RANDOM 8
  playerhit = playerhit + 2
  spelldamage = 1
END IF
```

Next add this block:

```
IF spelldamage = 1 THEN
  enemyHP = enemyHP - playerhit
  ALERT Your spell hits the enemy for $playerhit$ damage!
END IF

CLEAR TEXT
PRINT stats

IF enemyHP < 1 THEN
  ALERT You have defeated the enemy!
  GOTOCARD Victory Card
ELSE
  GOTOCARD Enemy's Turn Card
END IF
```

Treasure Chests

Requirements:
• *None*.

Create an Open Chest button and enter this in the button script:

```
IF chestempty = 0 THEN
  chestempty = 1
  chestgold = amount of gold in treasure chest
  gold = gold + chestgold
  ALERT You found $chestgold$ gold in the treasure chest!
```

ELSE
  ALERT The treasure chest is empty.
END IF

Weapons and Armor

Requirements:
• Basic Battle Engine

Enter this in the Player's Turn Card script in the battle engine:

playerhit = RANDOM base weapon damage
IF playerweapon$ = "weapon" THEN playerhit = playerhit + weapon strength

The code for armor is very similar to weapons. Enter this after the above code:

IF playerarmor$ = "armor" THEN enemyhit = enemyhit - armor strength
IF enemyhit < 0 THEN enemyhit = 0

Here are example weapon and armor blocks:

playerhit = RANDOM 5

IF playerweapon$ = "Sword" THEN playerhit = playerhit + 10
IF playerweapon$ = "Long Bow" THEN playerhit = playerhit + 15
IF playerweapon$ = "Magic Staff" THEN playerhit = playerhit + 20

IF playerarmor$ = "Leather Vest" THEN enemyhit = enemyhit - 5
IF playerarmor$ = "Plate Mail" THEN enemyhit = enemyhit - 10
IF playerarmor$ = "Dragon Hide" THEN enemyhit = enemyhit - 15

IF enemyhit < 0 THEN enemyhit = 0
enemyHP = enemyHP - playerhit