Slot Machine Overview

The slot machine client-side is composed of 4 components:

- HTML page
- CSS file, Images and sounds
- Javascript File
- Server-side component

and it has a number of optional features that you may want to keep or remove.

IMPORTANT NOTE: This is just the client-side package. If you have also purchased the server-side package, please read the README.PDF document in the server-side package before starting to work on these files, as some of these will be overwritten.

HTML Page

The HTML page that is included (index.php) has all the HTML elements to make the slot machine work, including all its options. You can remove the parts you don't need (see Optional Features section) and the rest should continue to work.

IMPORTANT: If you have purchased the server-side package too, you should overwrite this index.php page with the one in the server-side package. All these notes apply to both, but the server-side file is automatically integrated with the database configuration.

There are two main things you need to modify in this file:

1. The section at the top with basic configuration and information about the current user
    a. Number of icons per reel. By default, there are 6, but you can have a different number.
    b. Minimum and Maximum bet by the user: The slot machine includes an optional mechanism to let a user control his bet (how many credits he bets in every spin). This variable controls the maximum he can pick.

    c. Credits: How many credits the current user has.
    d. Day and Lifetime Winnings: Optional. You can keep track of what the user's winnings were this day, over his lifetime, or both. If you do, load them here, and they will be controlled by the rest of the code automatically.

    You modify all of these directly in the HTML file if you only have the client-side package. If you also have the server-side, refer to the server-side documentation on how to change these.

2. The Prizes Table
   You can render a prizes table showing which combinations give out which prizes. You need to do this yourself, since how you store this information can vary widely. At its most basic, you can just have the prizes be a part of the background image, and then you don't need to do anything, you can completely remove this section from the HTML. However, if you follow the pattern explained below, you get two features:

   - The prize just won gets highlighted

   For this to work, each line in your prize table must be a <div> with class "trPrize", and with an ID that identifies each prize. Each of your prizes has to have a specific ID (can simply be a number). The ID of the div must be: trPrize_[Prize_ID]. For example: trPrize_1, trPrize_2, trPrize_Laptop, etc. This prize ID (the second part of the div ID) will be returned by the server component to indicate which prize was won.

   - Payouts get multiplied by the bet automatically as it changes (if you allow bet changing)

   For this to work, the <div> needs to have another <div> inside, with class "tdPayout", that has inside in the HTML the actual payout, but also includes a data attribute specifying the number. If the value needs formatting to be applied, you can also specify a Prefix and a Suffix in data attributes. Example:

```
<div class="tdPayout" data-basePayout="100" data-payoutPrefix="$ "
     data-payoutSuffix=" --">$ 100 --</div>
```

The JS code will update the contents of the div when the user changes the bet using the prefix, the base payout number, the selected bet, and the suffix.

For this to work, prize payoutss need to be numeric. Read more about "types of prizes" in the Optional Features section.

The sample code that is provided to you loads a hard-coded list of prizes into a PHP array, and then has some generic code that renders that into HTML. We recommend you to keep this structure and simply modify the contents of the Prizes array, since that will make your life the easiest.

Read more about this in “Sprites for the Prizes table”

3. Optionally, you can remove the elements for features that you don't need.

CSS Files, Images and Sounds

To style your slot machine, you will start by changing the images, and then adapting the CSS file to modify the position and sizes of the HTML elements that are overlaid over these images. The slot machine comes with sample images and CSS. The more elements you can keep at the same size and relative positioning, the easier this will be.

Images

- **main_bg.png**: This is just a background image that doesn't have any components of the slot machine itself, it's just an image to show at the very back.
- **main_bg_machine.png:** This background contains all the "static" elements of the slot machine. You can have the prizes table here if you're not going to use the dynamic features mentioned above. You should remove anything that's not used from the image.
- **won_bg.png:** This background will be super-imposed over the main background when the user wins something. It will remain behind all other elements, though. You may have multiple of these if you want to show different colors/images when the user wins different prizes. More on this in the Optional Features section.
- **reel_strip.png**: This contains the icons in each reel. This image will repeat vertically, so it must be sized appropriately. The height of each element must be equal, and they must be centered around equally-separated imaginary guidelines. In the sample, we include a strip that you can use, and a Photoshop file with the guidelines to position elements. If you're going to use this size of reel, then each element is 120px tall, you should crop the file vertically at a multiple of 120.
- **reel_overlay.png**: This is a semi-transparent file that lays on top of the reels. We recommend leaving this as it is, since it is what gives the 3D effect that makes the reels look "round". You can scale this file if you need different sized reels, but the opacity / gradients are probably best left untouched.
- **spin_button.png:** Has the image of the button that users press in 4 states: normal, hover, pressed and disabled.
- **prize_won.png:** This image will be set as the background if the trPrize element in the prizes table when the user wins something. Each trPrize needs to be big enough to contain this entire image, and they must have no background in normal state. Their padding should be set so that contents are centered properly when this image is the background.
- **sprites.png:** Contains generic sprited images, like the bet up/down buttons and the sound on/off ones.
- **prizes_sprites.png:** Contains the images that will be show in the prizes table. Read more about this in the next section.

Sprites for the prizes table

The way the prizes table works and is rendered is designed to integrate easily with the prizes configuration in the server-side component. For each combination of images in the reels that would yield a certain prize, we would have one image in the **prizes_sprites.png** file.

If in the prize configuration the reel setting is just one image (1, 2, etc), then this is straightforward. However, we could have combinations such as “any blank”, or “images 1, 2 or 3”, etc. The server-side will generate strings for each of these.

For example:

- “1” turns into “prize_1”
- “1/2/3” turns into “prize_1slash2slash3”
- “*.5” turns into “prize_stardot5”

When rendering the table, each row will have one column per reel, and in those columns will be divs that have these strings as their CSS class.

You should make sure that for each of these you have an image in the **prizes_sprites.png** file, and a CSS rule pointing to the right coordinates.

For example, if you look at the template 1 included in this package:

- If you look at **reel_strip.png**, you’ll see the mapping between icons and numbers. 1 is banana, 5 is cherries, etc.
- If you look at the prize with a payout of 7, that will match either of the fruits. In the server, that is configured as “1/3/5”
- You will notice that that is rendered as a div with CSS class: prize_1slash3slash5
- If you look near the bottom of file template1.css, you will see the definition for that class:
    - #prizes_list .reelIcon.prize_1slash3slash5 { background-position: -0px -30px; }
- Finally, looking at **prizes_sprites.png**, in those coordinates, you see the image that will be rendered.

If you needed to add a new image for a new combination, for example, 1 and, 2, you would do the following:

- Create a new small image with a banana and a “red 7” in it. Add it to **prizes_sprites.png** (you would need to enlarge the image)
- Add a CSS rule for .prize_1slash2 pointing to the right coordinates in the prizes_sprites.png file.
- When rendering the prize in the prizes table, make sure the right <div> has that CSS class. (This will happen automatically if you are using the server-side component)

CSS File

The CSS file is separated into 2: slots.css, and template*.css.

Slots.css have general declarations that make the slot machine work. You will generally not need to modify this file very often.

Template*.css has the declarations that are specific to each design (positions, sizes, colors, images, etc)

Once you've replaced the images with your own, you'll need to update the Template CSS file included to fit the new dimensions / positions. I recommend reading the whole file once to have an idea of what it's trying to do, and modify as little as possible. Ideally, changing simply things like: left, right, top, bottom, width, height margins and paddings. Some of the slot machine logic is dependent on some settings in the CSS, so test thoroughly if changing anything other than these.

One specific change that must be done is in selector: "#slotMachineContainer #ReelContainer .reel". This specifies the height of the reel strip. You must set this to 3 times the height of the reel_strip.png image. Eg: the sample image is 720px, the height set in the sample CSS is 2160px.

Templates

The slot machine package comes with 5 templates bundled. You will notice that the index.php file includes stylesheets slots.css and template1.css. Changing the second one to template2.css, template3.css, etc will let you switch between templates.

If you are going to have a custom design, we suggest you copy the images directory for the template you are starting from, and also the CSS file, and make a new template, rather than modifying an existing one.

Sounds

The slot machine includes 3 sounds: spinning, payout and fast payout.

The spinning sound starts when the user hits spin, and includes the 3 "clicking" sounds for the reels. The click sounds are not played independently because the timing is never right, so they are included in the main sound, and Javascript animation needs to be timed to match the clicks in the spinning.mp3 file.

When the user wins a prize, the slot machine animates the increase in counters to make the payout moment last longer and be more fun. If the user has won a small prize (up to 80), payout.mp3 is played. If he's won more than 80, fast_payout.mp3 is played, and the counters increase much faster.

Javascript File

The Javascript file contains a number of configuration values at the top that you may or may not need to change. The rest of the file you should only need to change if you need specific custom features that the slot machine doesn't provide.

- **stripHeight**: The actual height of reel_strip.png
- **alignmentOffset**: This is an offset to get the items in the strip aligned perfectly.
- **firstReelStopTime**: Time since beginning of spin until the first reel stops spinning and starts bouncing.
- **secondReelStopTime**: Since first reel's stop time.
- **thirdReelStopTime**: Since second reel's stop time.
- **payoutStopTime**: Time since last reel starts bouncing until the couters start increasing and the winning sound plays. Ideally, make it a bit less than bounceTime, so the last reel is still bouncing when the coutners light up.

- **reelSpeedDifference**: If you want the different reels to move at different speeds, to look "more misaligned" set this value to a non-zero number.
- **reelSpeed1Delta, reelSpeed1Time, reelSpeed2Delta**: Reels may have to speed. They will move at reelSpeed1Delta speed for reelSpeed1Time milliseconds, and then at reelSpeed2Delta for the remaining time. I recommend leaving reelSpeed1Time as 0, and having only one speed, but you can make them go fast for a bit, and then slow down a bit before stopping.

- **positioningTime, bounceTIme, bounceHeight**: When the reel stops spinning, it goes into it's final "bounce" until it stops completely. It will move from wherever it is to bounceHeight pixels off of its final position in positioningTime milliseconds, and then it'll bounce around the end line for bounceHeight milliseconds. I recommend leaving these values alone, it's hard to get a better effect than the current one by tweaking them.

- **winningsFormatPrefix**: If the prizes are not just credits, then you can control their formatting with this. For example, if they are money, you can set this to "$ " or "£ ".

- **spinURL**: Points to the server endpoint to be called when spinning.
- **curBet**: Default starting bet
- **soundEnabled:** Whether your slot machine has sound.

Optional Features

There are a number of things that are easy to customize in this slot machine:

Types of prizes

The standard prize for slot machine is "credits" or "spins". You put money in, this allows you to get some spins, you spend those spins, and if you win, you get more spins, until you want to "cash out". This is the easier mode.

However, you may want to give users other types of prizes instead of, or in addition to credits. For example, you may want to give them free spins (say, 1 a day, or 10 a month), and as prize, money, or store credit. You may want to have "things" as prizes, like laptops. In this case, the "payout counter" doesn't work, it only works with numbers. You may also want to mix, and have some reel combinations give them more spins, and some others give them other things.

- Credits wins are controlled by the credits and payoutCredits fields of the server response. These must always be there.
- If your prize is just credits, or if you have separate prizes apart from credits, and you keep track of day/lifetime winnings, you can specify them in the dayWinnings, lifetimeWinnings and payoutWinnings fields. These will increase gradually just like credits do.
- You can show the last win too, which is just a string, so it can say anything like "laptop", "diamond", etc. This is specified in the lastWin field of the response.

Blank spaces between icons

You may have the reels only stop perfectly centered on the icons in the strip. However, you can only have them stop in the blank spaces between one icon and the next. You can do this by adding 0.5 to the reel position you return from the server.

Last win

Your server response may include a "lastWin" field which is a string that simply shows in HTML element "#lastWin" in each spin. If you won't use this, you don't need to return the lastWin field from server, and you can remove the #lastWin HTML element.

Day and Lifetime Winnings

If you want to show a counter with the winnings, separate from the credits counter, you can return elements dayWinnings, lifetimeWinnings and payoutWinnings. You can use one, the other, or both. The results will be shown, incrementing gradually when there was a win, in elements #dayWinnings and

#lifetimeWinnings respectively. If you won't use this, please still return the mentioned fields in your response, all set to 0. You can remove the HTML elements.

Also, "day" and "lifetime" winnings may also be used to represent other concepts as you choose, simply change the label in the HTML / Background image to say what these counters represent.

Changing Bets

You can allow the user to pick a bet between 1 credit and the maxBet field you specified in the HTML file. The bet will be sent up to the server with each spin request. If you don't want to use this feature, you can remove HTML elements #bet, #betSpinUp and #betSpinDown. The rest will work normally.

Sound

If your slot machine doesn't use sounds, you should remove element #soundOffButton.

Multiple "win" backgrounds

When the user wins something, #PageContainer gets added the class "won", which adds a second background that is lit up to show that the user has won. If the server response contains a field called "winType", then the value of "winType" will be added as a class, instead of "won". This allows you to show different backgrounds. You will need to add additional lines to the CSS file for each of these classes, similar to this one:

```
#PageContainer.won #PageContainerInner {
       background: transparent url(images/won_bg.png) 315px 0 no-repeat;
}
```

Different "reel" images

All the reels in the slot machine have the icons in the same positions in the reels. Some implementors don't like that. You can have 3 different strip orderings, by having 3 different images, and setting the backgrounds specifically for each reel in the CSS file. If you do this, however, the reel positions that you return from the server must be mapped to the new positions in this file. In other words, when you return reel position "N", that is the icon in the "N" position of that reel's background, it's not the "nth" icon. The slot machine doesn't know which icon is where, it assumes they are all in the same place.

Server-side component

IMPORTANT: The rest of this document is only relevant if you are writing your own server-side, or it you want to modify it to add new features. If you bought the server-side component, you can skip this section.

When the user spins, the Javascript sends a POST request to the server, to the URL specified at the top of the JS file, with 2 elements of data: **bet** and **windowID**

The bet is how many credits the user decided to use in this spin. You need to take this into account when subtracting from their balance, and when returning the payout values, they need to be scaled appropriately by the server.

The windowID can be used to keep track of windows if the user opens multiple browser windows at once. I'm just identifying each one with a random number, but you may use a more sophisticated system, among other things, to detect the situation and warn them that they can't do that.

Validation

On the server, you should check that the request is valid. At the very least, that the user is still logged in, and that they have enough credits to do the spin.

If things are not well, you should reply with a valid JSON response with two fields:

- **success:** set to false
- **error:** If error is the value "loggedOut", then the HTML element #loggedOutMessage will be shown. If not, an alert will pop-up with the contents of the **error** field.

If the request fails in any way, like not being able to connect, returning a 500 status code, returning a non-valid JSON response, etc, HTML element #failedRequestMessage will be shown to the user.

If either of these situations happen, the slot machine will be disabled until the user reloads the page.

Performance

For the best possible user experience, the reels start spinning immediately upon pressing "Spin", instead of waiting for the server response. However, the reels cannot stop spinning until a response is received.

Since the clicking sounds of the reels stopping are hard-coded into the background MP3 file, the server must respond before the first reel needs to stop, or the sound will be out of sync. With the default sound, this is about 500ms.

If the server never responds, the reels will seem to "keep spinning forever". They will actually stop on the 10 second timeout, but users will report "forever".

Response

The response from the server must be a valid JSON response, with the following fields:

- **success:** (required) true or false
- **error:** (required if success is not true) see "Validation" above for a description.
- **reels:** (required) specifies the position of each reel. It's an array with 3 elements, each of then a number that can either be an integer indicating which icon to have in the result line (1-based). If the number ends in 0.5, it represents the blank space between an icon and the next. So, 2 is the 2nd icon in reel_strip.png, while 2.5 is the blank space between icons 2 and 3.
- **credits:** (required) the number of credits available to the user after the spin, including however many credits he may have won.
- **dayWinnings**: (required) The total winnings of the user today. Send 0 if you're not using this feature.
- **lifetimeWinnings**: (required) The total winnings of the user in his lifetime. Send 0 if you're not using this feature.

- **prize:** (required) null if the user didn't win. If not null, the slot machine shows the win state. **prize** is an object with multiple fields:
  - **id:** (required) the ID of the prize won. Used to highlight the right row in the prizes table. Send it anyway even if you're not using that feature.
  - **payoutCredits**: (required) how many credits the user just won. This is technically redundant, but we send both this and **credits** because we don't trust the state of the client. The counter will count up from *(credits - payoutCredits)* to *credits*, ignoring how many credits the client "thought it had".
  - **payoutWinnings**: (required) If you're using dayWinnings or lifetimeWinnings, how much the user won in this spin (equivalent to payoutCredits but for the Winnings elements). Send 0 if you're not using this feature.
  - **winType**: (optional) If you have different backgrounds for the win state, send this. The value of this field will be added as CSS class to #PageContainer

- **lastWin**: (optional) string to show in the #lastWin element.