jQuery: Eigenes Plugin [Part 1: Grundlagen]

== Einführung ==Hallo. Hier erkläre ich kurz, wie man sein eigenes jQuery-Plugin erstellen und anschließend auch benutzen kann. Das Beispiel, das ich verwenden werde ist nicht sehr sinnvoll, da es nur zur Demonstration dient. Es wird erklärt, wie man ein ganz einfaches Plugin erstellt und anschließend noch, wie man dieses Plugin mit eigenen Parametern verseht. Ihr solltet grundlegendes Wissen von jQuery besitzen.
== Anfang ==Zu aller erst erstellen wir eine HTML-Seite mit dem Standardinhalt:

Quellcode

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <title>Eigenes Plugin</title>
5. </head>
6. <body>
7. </body>
8. </html>
```

Alles anzeigen

Nun erstellen wir eine leere Javascript-Datei, speichern sie unter dem Namen "plugin_first" ab und binden sie ein. Natürlich dazu auch noch jQuery über Google. Nun fehlt uns noch eine Box, auf die wir unser Plugin verwenden. Dieser div geben wir die id "test_box", die Farbe rot, die Maße 100px * 100px und die Position absolute, da wir die Box später animieren werden. Der Code unserer HTML-Seite sollte nun in etwa so aussehen:

Quellcode

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <title>Eigenes Plugin</title>
5. <script language="javascript" type="text/javascript"
   src="https://ajax.googleapis.com/ajax/libs/jquery/1.6.4/jquery.min.js"></script>
6. <script language="javascript" type="text/javascript" src="plugin_first.js"></script>
7. <style type="text/css">
8. <!--
9. #test_box {
10. width: 100px;
11. height: 100px;
12. background-color: red;
13. position: absolute;
14. }
15. -->
16. </style>
17. </head>
18. <body>
19. <div id="test_box"></div>
20. </body>
21. </html>
```

Alles anzeigen

== Plugin erstellen ==Gehen wir zuerst zu unserer plugin_first-Datei. Um ein eigenes Plugin zu erstellen, beginnen wir mit $.fn.extend. Dies gibt praktisch bescheid, dass wir ein eigenes Plugin schreiben.

plugin_first.js
Quellcode

```
1. $.fn.extend({
2. });
```

Unsere eigene Funktion wird pluginOwn heißen. Um unser Plugin zu erstellen und zu benennen, machen wir folgendes:
Quellcode

```
1. $.fn.extend({
2. pluginOwn: function()
3. {
4. }
5. });
```

Um zu erreichen, dass auf jedes Element, das später ausgewählt wird die Funktion ausgeführt wird, schreiben wir zu aller erst Folgendes in unser Plugin:
Quellcode

```
1. $.fn.extend({
2. pluginOwn: function()
3. {
4. return this.each(function()
5. {
6. //Hier kommt unser Plugin rein
7. });
8. }
9. });
```

Alles anzeigen

== Funktion des Plugins ==Als Beispiel nutzen wir jetzt einfach die animate Funktion, es kann aber auch jede andere jQuery- oder Javascript-Funktion benutzt werden.

plugin_first.js
Quellcode

```
1. $.fn.extend({
2. pluginOwn: function()
3. {
4. return this.each(function()
5. {
6. $(this).animate({top: '+=100px'}, 1000);
7. });
8. }
9. });
```

Alles anzeigen

Durch das *$(this)* wird das Element ausgewählt, das wir später mit der Funktion auswählen werden. Der eigentliche Effekt ist *animate({top: '+=100px'}, 1000)*. Das Element wird um 100 Pixel nach unten verschoben, dies geschieht mit einer

Verzögerung von einer Sekunde.

== Testen ==Nun wollen wir das Plugin testen. Dazu fügen wir nach unserem style-Teil auf der HTML-Seite folgendes ein:

Quellcode

```
<script language="javascript" type="text/javascript">
$(document).ready(function()
{
$('#test_box').pluginOwn();
});
</script>
```

Dies sollte sich von alleine erklären.Jetzt kann man die Datei im Browser öffnen und schaut da, die Box bewegt sich.

== Parameter übergeben ==Man sieht ja bei jeder Funktion, dass auch Parameter übergeben werden. Dazu gehen wir wieder zu unserer Javascript-Datei.

Quellcode

```
$.fn.extend({
pluginOwn: function(pixel, zeit) //Diese Zeile
{
return this.each(function()
{
$(this).animate({left: '+='+pixel+'px'}, zeit);
});
}
});
```

Alles anzeigen

Nach unsererm Namen schreiben wir in die Funktionsklammern, die jeweiligen Variablennamen, die wir mit der Funktion übergeben wollen. In unserem Fall um wie viele Pixel sich die Box nach rechts bewegt und wie lange sie dazu braucht. Anschließend werden diese Variablen in die Funktion eingesetzt.
Um dies jetzt zu testen, müssen wir unsere HTML-Datei auch noch anspassen. Wir schreiben dort in die Funktion, dass die Box sich um 250 Pixel verschieben soll und dafür 1250 Millisekunden brauch.

plugin_first.html

Quellcode

```
//ersetze:
$('#test_box').pluginOwn();
//durch:
$('#test_box').pluginOwn(250, 1250);
```

== Fertig ==Eure Dateien sollten nun so aussehen:

plugin_first.js

Quellcode

```
$.fn.extend({
pluginOwn: function(pixel, zeit)
{
return this.each(function()
```

```
{
$(this).animate({top: '+='+pixel+'px'}, zeit);
});
}
});
```

Alles anzeigen

HTML-Seite

Quellcode

```
<!DOCTYPE html>
<html>
<head>
<title>Eigenes Plugin</title>
<script language="javascript" type="text/javascript"
src="https://ajax.googleapis.com/ajax/libs/jquery/1.6.4/jquery.min.js"></script>
<script language="javascript" type="text/javascript" src="plugin_first.js"></script>
<style type="text/css">
<!--
#test_box {
width: 100px;
height: 100px;
background-color: red;
position: absolute;
}
-->
</style>
<script language="javascript" type="text/javascript">
$(document).ready(function()
{
$('#test_box').pluginOwn(250, 1250);
});
</script>
</head>
<body>
<div id="test_box"></div>
</body>
</html>
```

Alles anzeigen

== Schlusswort: ==Hier gehts zum zweiten Teil: jQuery: Eigenes Plugin [Part 2: Erweiterte Paramter]