Jquery Tutorial: Binding to future events - Developer News

NEWS_PDF_AUTHOR: kaotik

NEWS_PDF_DATE: 2010/4/1 13:30:00

This tutorial will teach you the benefits of $.live() and how to pass a smarty template with ajax.

What is future binding?
This is relatively a new problem created with the advent of ajax calls. The best way to understand it is with a practical example:

Starting simple.
Download my tutorial module (tutorial17.zip) and install it.

Now go to the main page and click on tutorial.
You will now see the words: "Click Here". When you do an alert window pops up with "Hello world". This is very simple jquery code that I've explained in previous tutorials.

Now let's bring some ajax into this.
In the tutorial folder create a file called process.php and place this inside the file: "This was passed using ajax."
Now go to tutorial/templates/tut_main.html and replace all code with this:

```
$(document).ready(function(){

$('#clickme').click(function() {
  $('#ajxDiv').load('/modules/tutorial/process.php');
});
}); //END READY
```

```
Click here
```

```
 
```

Now test it. As you can see it's a simple ajax call.
Caution: For ajax to work properly, turn off xoops debug or set it to a popup window.

Passing smarty templates
Let's take this a step further and instead of passing some text through ajax, let's pass a smarty template.

Open process.php and replace everthing with this:

Open file tutorial/templates/list.html and replace everything with this:

```
list 1
list 2
list 3
```

Test the module. You now have a smarty template being passed through a ajax call.

Binding to ajax template
Now let's try binding a click function to the loaded template.
Open tutorial/templates/tut_main.html and replace all code with this:

```
$(document).ready(function(){

$('#clickme').click(function() {
  $('#ajxDiv').load('/modules/tutorial/process.php');
});

$('#myList').click(function() {
alert("hello world");
});

}); //END READY
```

```
Click here
```

```
 
```

The intended result is that, when you click on an item of the list, an alert window should pop up.... Try it.
Nothing happens.

The reason for this is that the DOM has already been assembled and all javascript executed. Using ajax, you are passing html after everything has been assembled. In practical terms this means you can't bind to future events (such as links in a template) using the normal methods. Thankfully the people over at jquery are highly resourceful and have provided a method to deal with this.

$.live()
Open file tutorial/templates/tut_main.html and replace all code with this:

```
$(document).ready(function(){

$('#clickme').click(function() {
  $('#ajxDiv').load('/modules/tutorial/process.php');
});

$('#myList').live('click', function() {
alert("hello world");
});

}); //END READY


Click here


 
```

Now try running the module. It works! Isn't jquery grand!

I am now using .live with an event handler attached (click).
live() is very flexible and gives you the freedom to attach javascript alongside ajax calls.

Let me know if this was helpful.

This tutorial will teach you the benefits of $.live() and how to pass a smarty template with ajax.

What is future binding?
This is relatively a new problem created with the advent of ajax calls. The best way to understand it is with a practical example:

Starting simple.
Download my tutorial module (tutorial17.zip) and install it.

Now go to the main page and click on tutorial.
You will now see the words: "Click Here". When you do an alert window pops up with "Hello world". This is very simple jquery code that I've explained in previous tutorials.

Now let's bring some ajax into this.
In the tutorial folder create a file called process.php and place this inside the file: "This was passed using ajax."
Now go to tutorial/templates/tut_main.html and replace all code with this:

```
$(document).ready(function(){

$('#clickme').click(function() {
  $('#ajxDiv').load('/modules/tutorial/process.php');
});
}); //END READY
```

```
Click here
```

```
 
```

Now test it. As you can see it's a simple ajax call.
Caution: For ajax to work properly, turn off xoops debug or set it to a popup window.

Passing smarty templates
Let's take this a step further and instead of passing some text through ajax, let's pass a smarty template.

Open process.php and replace everthing with this:

Open file tutorial/templates/list.html and replace everything with this:

```
list 1
list 2
list 3
```

Test the module. You now have a smarty template being passed through a ajax call.

Binding to ajax template
Now let's try binding a click function to the loaded template.
Open tutorial/templates/tut_main.html and replace all code with this:

```
$(document).ready(function(){

$('#clickme').click(function() {
  $('#ajxDiv').load('/modules/tutorial/process.php');
});

$('#myList').click(function() {
alert("hello world");
});

}); //END READY
```

```
Click here
```

```
 
```

The intended result is that, when you click on an item of the list, an alert window should pop up.... Try it.
Nothing happens.

The reason for this is that the DOM has already been assembled and all javascript executed. Using ajax, you are passing html after everything has been assembled. In practical terms this means you can't bind to future events (such as links in a template) using the normal methods. Thankfully the people over at jquery are highly resourceful and have provided a method to deal with this.

$.live()
Open file tutorial/templates/tut_main.html and replace all code with this:

```
$(document).ready(function(){

$('#clickme').click(function() {
  $('#ajxDiv').load('/modules/tutorial/process.php');
});

$('#myList').live('click', function() {
alert("hello world");
});

}); //END READY


Click here


 
```

Now try running the module. It works! Isn't jquery grand!

I am now using .live with an event handler attached (click).
live() is very flexible and gives you the freedom to attach javascript alongside ajax calls.

Let me know if this was helpful.