



AcroTeX.Net

AcroTeX PDF Blog

Processing Acrobat Forms using JavaScript

External Processing of a Field

Part 1: The Basics

D. P. Story

1. Introduction

The Acrobat forums are filled on a daily basis with questions on Acrobat form fields and how to process them, that is, how to extract the value of the field, or to change the appearance of the field. We begin a series of articles on my own understanding of this topic.

In my view, there are two major cases:

1. A script *external* to the target field (perhaps another field) accesses the field.
2. A script *internal* to the target field processes the data input into it, or accesses the current value, or changes its appearance.

In this blog, we take up the first case, the case of an external script accessing the field; this is the easiest case.

Of course, the document that guides us in our efforts is the [JavaScript for Acrobat API Reference](#)¹. This is the Version 9 document and is available in html form only. To find this document on the Adobe web site pages, go to the [Acrobat Developer Center](#), click the DOCUMENTATION tab, then click the [Acrobat 9 SDK Documentation is available online and for offline use](#) link. After the livedocs page is loaded, drill into the JavaScript topic in the left navigation panel; you'll see [Developing Acrobat Applications Using JavaScript](#) and [JavaScript for Acrobat API Reference](#). Yes, Adobe has hidden these documents very nicely. The former document is also a useful resource, though I will not consult it, I promise.

If you want a PDF version of the [JavaScript for Acrobat API Reference](#), you must use the Version 8 edition, available at the [Acrobat Developer Center](#).² (Click the link JavaScript for Acrobat in the right navigation panel.) We will not be using anything from Version 9 in this series of articles—but who knows.

An excellent Internet reference to core JavaScript can be found at the [Mozilla Developer Center](#).³

Throughout this topic thread, we shall be concerned with

- Text Fields
- Button Fields
 - (a) Push Button
 - (b) Check Box
 - (c) Radio Button
- Choice Fields
 - (a) List Box
 - (b) Combo Box
- Signature Fields

Signature Fields are included in the list for completeness, but this type of field will not be covered in these articles, hence, it appears grayed out. We take up each of the others in turn.

¹http://livedocs.adobe.com/acrobat_sdk/9/Acrobat9_HTMLHelp/index.html

²http://www.adobe.com/go/acrobat_developer

³<https://developer.mozilla.org/en/JavaScript>

2. The Basics

Basic to processing a field is getting its Field object, and triggering JavaScript to perform the desired tasks.

2.1. Acquiring the Field object

Keeping in mind that we want to process the target field using a script that is external to that field, the first step is to acquire the Field object of the target field. The most frequently used methods for doing this is the `Doc.getField()` method. Assuming we are processing a target field in the same document as the executing script, we would use

```
var f = this.getField(fieldname);
```

where *fieldname* is the name of the target field, and is a string type. The value of the JavaScript variable `f` contains the Field object for the field whose name is *fieldname*.

Sometimes, you are not sure that the field exists, if the field does not exist, the `Doc.getField()` method returns a null value. You can test for null as follows:

```
var f = this.getField(fieldname);
if ( f == null ) app.alert("Field does not exist!");
else {
    Process the field
}
```

After having acquired the Field object of the target field, you can use the properties and methods of the Field object to process the target field.

2.2. Executing JavaScript

A few brief remarks on executing JavaScript. For the time being, we will use the user interface terminology. All field have a properties dialog box, and an Actions tab. The Action tab features a Select Trigger option list and a Select Action option list. These articles concern exclusively the Run a JavaScript action found in the Select Action option list. The JavaScript is run by a trigger; common triggers are Mouse Up, Mouse Down, Mouse Enter, Mouse Exit, On Focus, and On Blur.

For the Mouse Up trigger, for example, when the user releases the left mouse button while the mouse is over the bounding rectangle of the field, the Mouse Up trigger is tripped, and any action associated with that event is executed.

There are other triggers as well, depending on the form field, but the ones listed above will serve us well for now.

To illustrate the points made in this section, the following simple example is offered:

Example 2.1. From a button, acquire the current value of a text field and display the result in an alert box.

Enter text into the field:

The verbatim listing of the JavaScript executed off a Mouse Up trigger of the push button is given below:

```
1 var f = this.getField("myTxt2-1");
2 var v = f.value;
3 var stripv = v.replace(/\s*/g, "");
4 if ( stripv == "" )
5     app.alert("Nothing but white space in the field.");
6 else
7     app.alert("You entered \"" + v + "\"" into the text field!");
```

Comments. Get the Field object of the target field (1); get the value of the target field (2) using the `value` property of the Field object; use a regular expression to strip out all white space (3) and save the result as `stripv`; (4) if `stripv` is an empty string, the user is trying to fool us, we let him have it (5), otherwise we report the result (7).

In line (1), note that the field name is enclosed in double quotations, when the field name is a string literal. When the field name is passed using a JavaScript variable, we simply use the variable, already a string type; for example,

```
var fname = "myTxt2-1";
var f = this.getField(fname);
...
```

□

Important: JavaScript variables declared as actions for a field trigger are not scoped locally and are known throughout the document. Other scripts may use the variable `f` or `v`; care must be made to give these variables their own values, for otherwise, they may take on values acquired earlier by the executing of Field script, or some other script.

That's pretty much it for now, I simply must get back to my retirement. ☹