

ServiceNow 201:

7 - GlideAjax



Course Outline

1 Course Introduction

2 Development Overview

3 Scripting Locations

4 GlideRecord

5 GlideSystem

6 GlideForm & GlideUser

7 GlideAjax

8 Exploring Other APIs

9 Becoming A Scripting Master

10 Creating A Custom App

Section Outline

1 GlideAjax Introduction

2 AJAX

3 Synchronous vs Asynchronous

4 GlideAjax Stages

5 Example: GlideAjax Process

6 Show Me The Code!

7 Revisiting Script Includes

8 The JavaScript Callback

9 JSON vs XML

10 GlideAjax API Overview & Methods

11 GlideAjax Demo

12 Section Recap

GlideAjax Introduction



- Client-side
- Used to call server-side Script Includes
- Similar to jQuery's ajax method



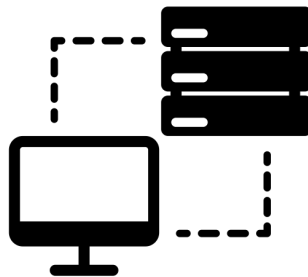
The GlideAjax class enables a client script to call server-side code in a script include.

[ServiceNow Docs](#)



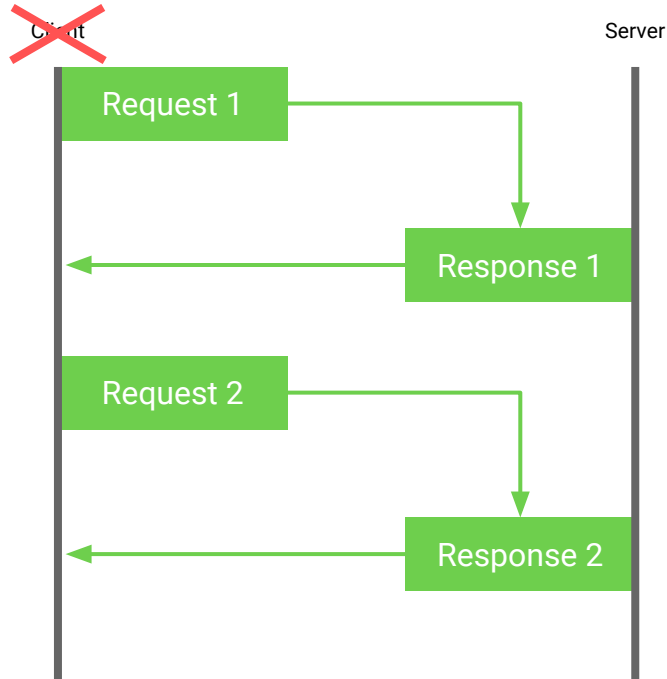
AJAX

- Asynchronous JavaScript and XML
- Popularized by Google in Google Maps and Gmail
- Way to make client-side requests to server-side without requiring a “page reload”
- Uses browser’s XMLHttpRequest API

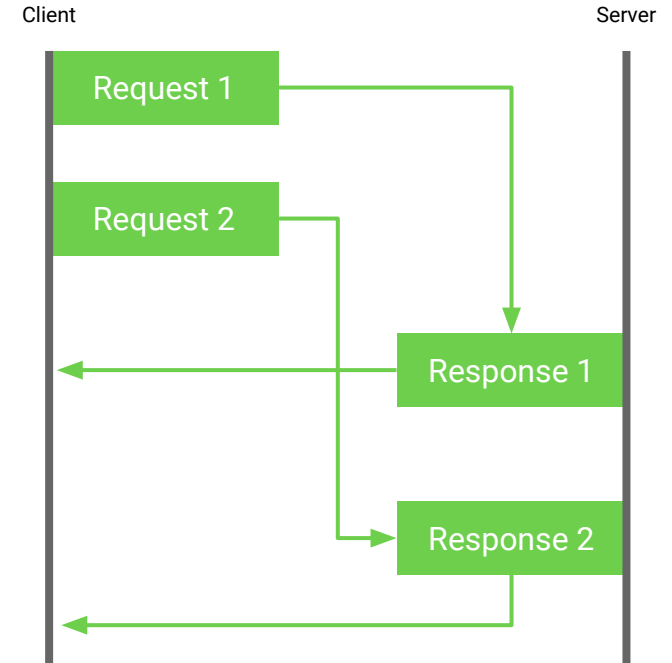


Synchronous Versus Asynchronous

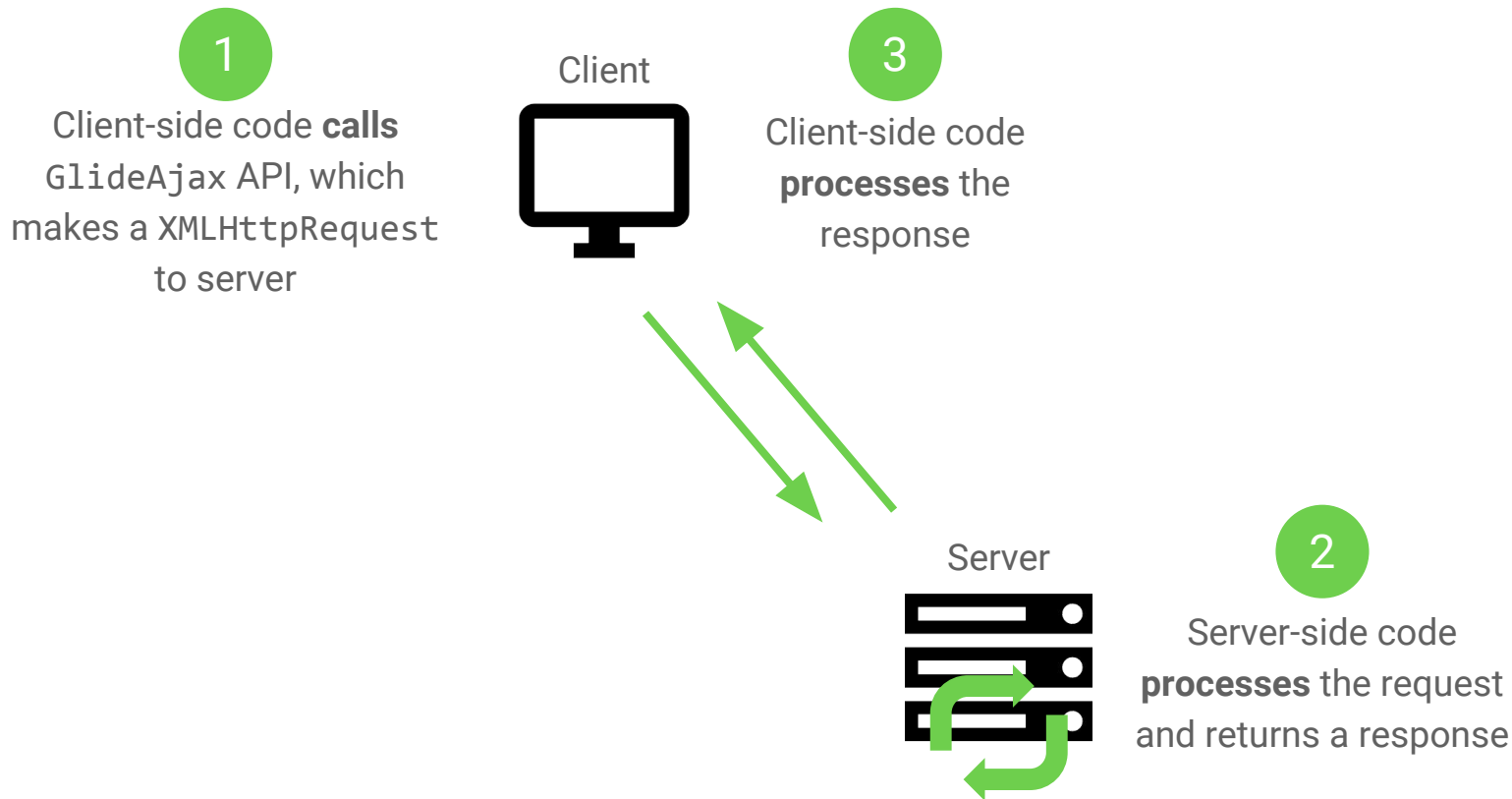
Synchronous



Asynchronous



3 Stages of GlideAjax



2 Scripting Locations To GlideAjax

1

Client-side code

Client



Client Script
UI Page

2

Server-side code

Server



Script Include



Client script

```
1 var ga = new GlideAjax('ServiceNow201GlideAjax');
2 ga.addParam('sysparm_name', 'getLatestIncidents');
3 ga.addParam('sysparm_limit', '5');
4 ga.getXML(ajaxProcessor);
5
6 function ajaxProcessor(response) {
7     console.log('Response payload: ' + response);
8     var answer = response.responseXML.documentElement.getAttribute('answer');
9     console.log('Answer: ' + answer);
10    var json = answer.evalJSON();
11    console.log('JSON: ' + json);
12    console.log(json[0].shortDescription);
13 }
```

Name Application

API Name Accessible from

Client callable ☒ Active ☒

Description

Script

```
1 var ServiceNow201ScriptInclude = Class.create();
2 ServiceNow201ScriptInclude.prototype = Object.extend(Object.prototype, {
3     getIncidentStatus: function() {
4         var incidentNumber = this.getParameter('sysparm_incident_number');
5         if(!incidentNumber) {
6             var incidentGR = new GlideRecord('incident');
7             incidentGR.get('number', incidentNumber);
8             return incidentGR.state.getDisplayValue();
9         } else {
10            return 'No incident was found';
11        }
12    },
13    type: 'ServiceNow201ScriptInclude'
14 });
```

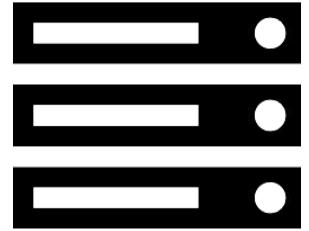

Example: GlideAjax Process

1

Client makes request for a page that contains a Client Script with GlideAjax



Task form



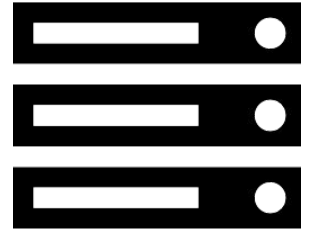
Example: GlideAjax Process

2

Server sends client task form data along with Client Script



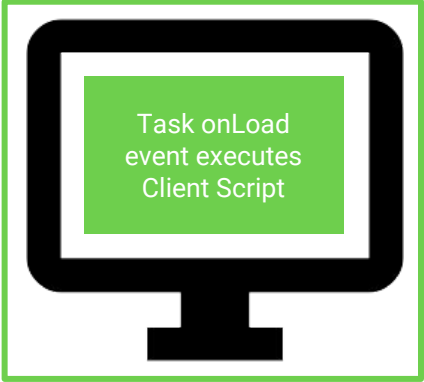
Task data w/ Client Script



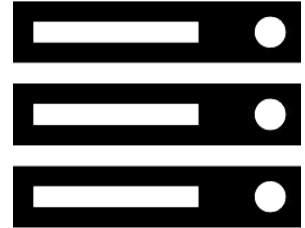
Example: GlideAjax Process

3

After an onLoad event occurs, the client side script is executed which calls the GlideAjax API



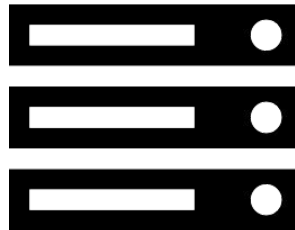
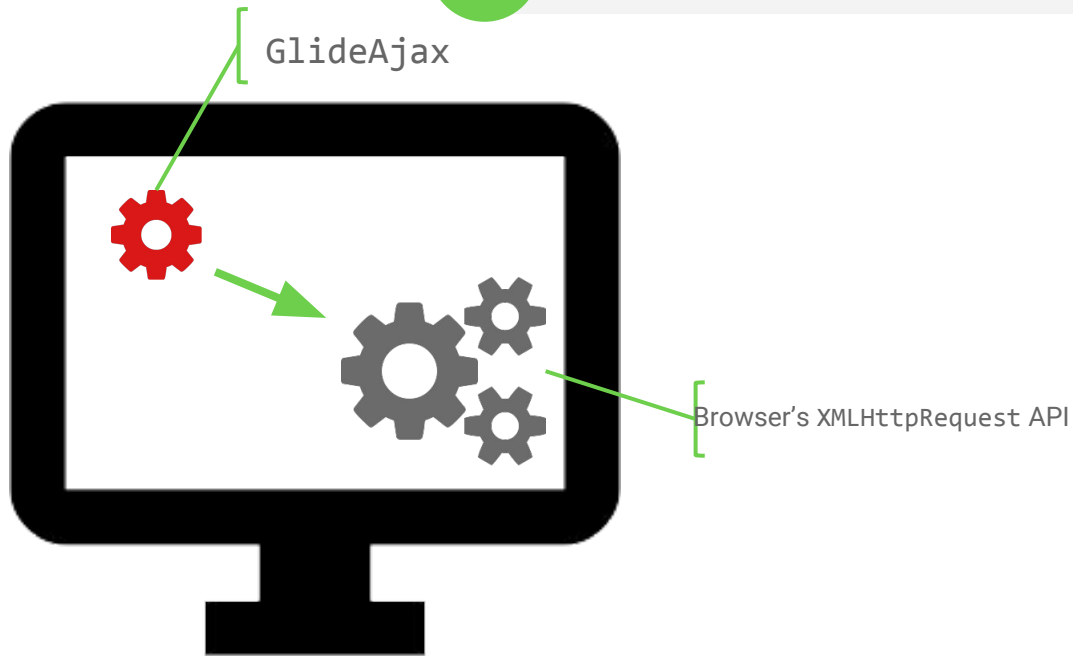
Task onLoad
event executes
Client Script



Example: GlideAjax Process

4

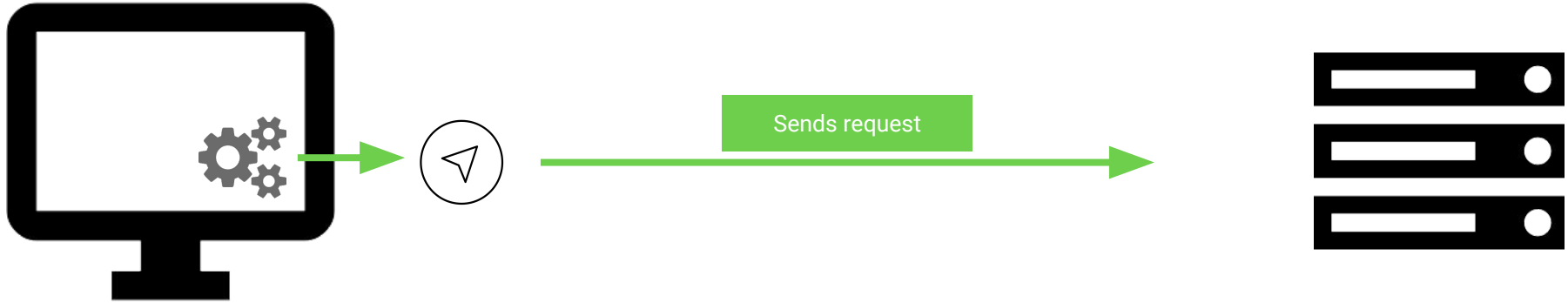
GlideAjax accesses browser's XMLHttpRequest API and generates a request



Example: GlideAjax Process

5

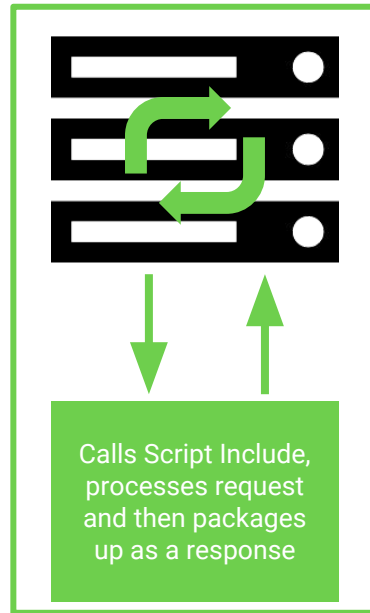
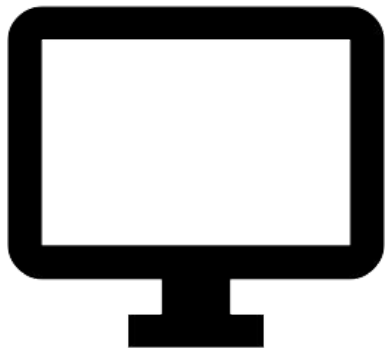
Browser's XMLHttpRequest API sends geolocation data back to ServiceNow in the background



Example: GlideAjax Process

6

Request from client invokes Script Include, where request data is used to call specific methods with arguments. Then data is packaged up in the form of a response.



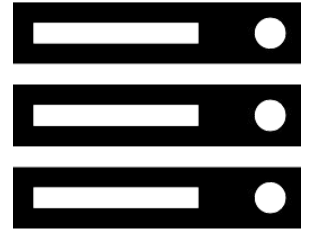
Example: GlideAjax Process

7

Browser receives response from server side



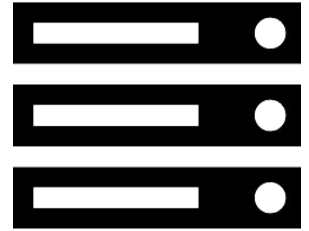
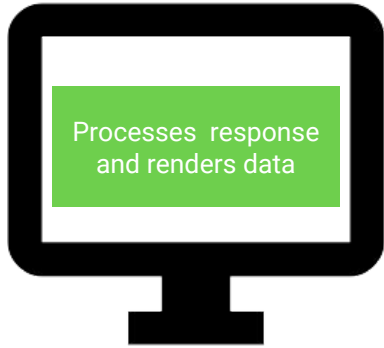
Sends response



Example: GlideAjax Process

8

Client Script callback processes returned data and updates location field on task



Show Me The Code!

- Update the short description field of an incident to *Hello world!* on-load

1 Client-side code



2 Server-side code



```
1 <xml
2   answer="Hello world!"
3   sysparm_max="15"
4   sysparm_name="sayHello"
5   sysparm_processor="ServiceNow201GlideAjax" />
```

```
1 ▾ function onLoad() {
2   //Type appropriate comment here, and begin script below
3   var ga = new GlideAjax('ServiceNow201GlideAjax');
4   ga.addParam('sysparm_name', 'sayHello');
5   ga.getXML/ajaxProcessor);
6 }
7
8 ▾ function ajaxProcessor(response) {
9   var answer = response.responseXML.documentElement.getAttribute('answer');
10  g_form.setValue('short_description', answer);
11 }
```


```
1 ServiceNow201GlideAjax = Class.create();
2 ServiceNow201GlideAjax.prototype = Object.extend(Object.prototype, {
3
4   sayHello: function() {
5     return 'Hello world!';
6   },
7
8   type: 'ServiceNow201GlideAjax'
9 });
```

It All Starts With GlideAjax

1

1. Create a new GlideAjax object
2. Add name of Script Include method as sysparm_name parameter
3. Call getXML() method and pass the name of the callback as an argument

```
1 ▼ function onLoad() {  
2     //Type appropriate comment here, and begin script below  
3     var ga = new GlideAjax('ServiceNow201GlideAjax');  
4     ga.addParam('sysparm_name', 'sayHello');  
5     ga.getXML-ajaxProcessor);  
6 }  
7  
8 ▼ function ajaxProcessor(response) {  
9     var answer = response.responseXML.documentElement.getAttribute('answer');  
10    g_form.setValue('short_description', answer);  
11 }
```

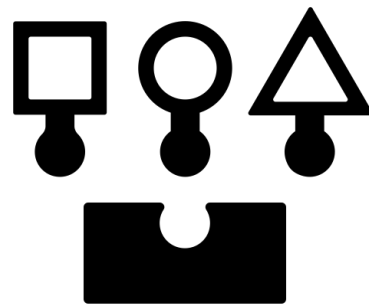


Note: Start all parameter names with sysparm_

Revisiting Script Includes

2

- Run on the server-side
- Contain reusable snippets of code, making them modular
- Only executed when invoked from another source
 - Client-side using GlideAjax
 - Server-side using the new JavaScript operator
- May extend other JavaScript classes
- When used for GlideAjax
 - Must have **Client callable** set to true
 - Extends the AbstractAjaxProcessor class
 - Typically queries a table and returns record(s) to client-side as JSON
 - Has access to any variable from client-side that starts with sysparm_
 - Uses sysparm_name to invoke method in Script Include



Extending AbstractAjaxProcessor

2

- AbstractAjaxProcessor is an out-of-the-box Script Include
- Provides helper methods
- Client callable checkbox automatically generates required JavaScript

```
1 var ServiceNow201GlideAjax = Class.create();
2 ServiceNow201GlideAjax.prototype = Object.extend(AbstractAjaxProcessor, {
3
4     sayHello: function() {
5         return 'Hello world!';
6     },
7
8     type: 'ServiceNow201GlideAjax'
9 });
```

Name	AbstractAjaxProcessor	Application
API Name	global.AbstractAjaxProcessor	Accessible from
Client callable	<input checked="" type="checkbox"/>	Active
Description	Base ajax processor class that other ajax processors extend	
Script	<pre>1 // Base ajax processor class that other ajax processors extend 2 // 3 // note that some methods return Java values, not JavaScript values 4 5 var AbstractAjaxProcessor = Class.create(); 6 7 AbstractAjaxProcessor.prototype = { 8 CALLABLE_PREFIX : 'ajaxFunction_', 9 10 initialize : function(request, responseXML, gc) { 11 this.request = request; 12 this.responseXML = responseXML;</pre>	

The GlideAjax Script Include

2

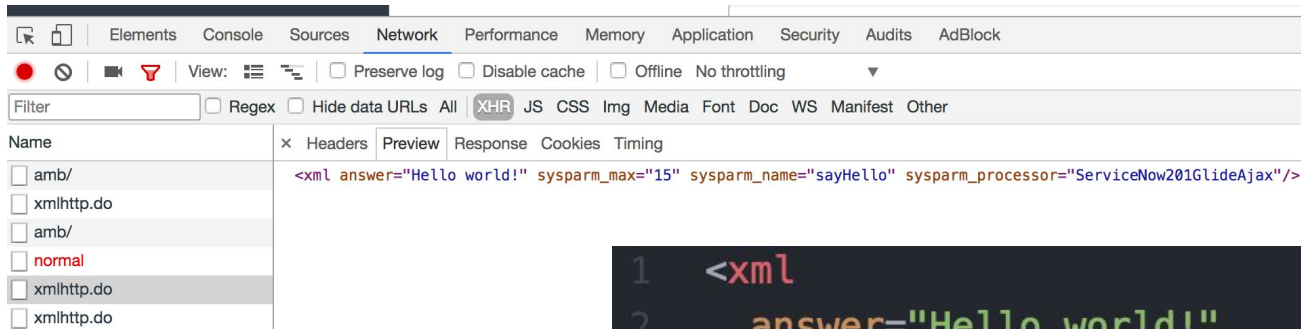
1. Method passed in sysparm_name gets invoked
2. Server-side scripts are ran
3. return statement ends server-side script execution
4. Response is packaged up as XML and sent to client

```
var ga = new GlideAjax('ServiceNow201GlideAjax');  
ga.addParam('sysparm_name', 'sayHello');  
ga.getXML/ajaxProcessor);
```

```
1 var ServiceNow201GlideAjax = Class.create();  
2 ServiceNow201GlideAjax.prototype = Object.extend(Object.prototype, {  
3  
4 sayHello: function() {  
5     return 'Hello world!';  
6 },  
7  
8 type: 'ServiceNow201GlideAjax'  
9 });
```

Returned Payload

- XML is returned to the client-side from server-side (even if JSON)
- Use the following to retrieve *answer*
 - `response.responseXML.documentElement.getAttribute('answer')`



```
1 <xml
2   answer="Hello world!"
3   sysparm_max="15"
4   sysparm_name="sayHello"
5   sysparm_processor="ServiceNow201GlideAjax"/>
```

The JavaScript Callback

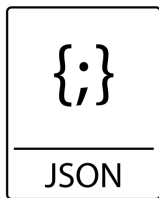
- A callback is a function that is passed as an argument, which is then executed at a later time
- Once the client receives the response from the server, the callback is invoked

```
1  function onLoad() {  
2      //Type appropriate comment here, and begin script below  
3      var ga = new GlideAjax('ServiceNow201GlideAjax');  
4      ga.addParam('sysparm_name', 'sayHello');  
5      ga.getXML/ajaxProcessor);  
6  }  
7  
8  function ajaxProcessor(response) {  
9      var answer = response.responseXML.documentElement.getAttribute('answer');  
10     g_form.setValue('short_description', answer);  
11 }
```

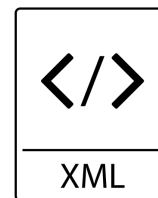


Dive Deeper: For more information on JavaScript callbacks, checkout [this article](#)

JSON Or XML?



- **JavaScript Object Notation**
- Very easy & fast to parse
- Uses collections of key/value pairs
- Supports arrays



- **Extensible Markup Language**
- Uses nodes & node attributes
- Can be a pain to parse



Dive Deeper: Checkout [this YouTube video](#) for a deeper understanding on XML vs JSON

XML Versus JSON

XML

```
1 // XML
2 <incidents>
3   <incident>
4     <number>INC00001</number>
5     <category>Test</category>
6     <priority>1</priority>
7     <state>new</state>
8   </incident>
9   <incident>
10    <number>INC00002</number>
11    <category>Test 2</category>
12    <priority>2</priority>
13    <state>active</state>
14  </incident>
15  <incident>
16    <number>INC00003</number>
17    <category>Test 3</category>
18    <priority>3</priority>
19    <state>resolved</state>
20  </incident>
21 </incidents>
```

JSON

```
1 // JSON
2 {
3   "incidents": [
4     {
5       "number": "INC00001",
6       "category": "Test",
7       "priority": "1",
8       "state": "new"
9     },
10    {
11      "number": "INC00002",
12      "category": "Test 2",
13      "priority": "2",
14      "state": "active"
15    },
16    {
17      "number": "INC00003",
18      "category": "Test 3",
19      "priority": "3",
20      "state": "resolved"
21    }
22  ]
23 }
```

GlideAjax getXMLAnswer() Method

- Shortcut to
 - getXML()
 - response.responseXML.documentElement.getAttribute('answer')

```
1 ▼ function onLoad() {  
2     var ga = new GlideAjax('ServiceNow201GlideAjax');  
3     ga.addParam('sysparm_name', 'sayHello');  
4     ga.getXMLAnswer(ajaxProcessor);  
5 }  
6  
7 ▼ function ajaxProcessor(answer) {  
8     //var answer = response.responseXML.documentElement.getAttribute('answer');  
9     g_form.setValue('short_description', answer);  
10 }
```

GlideAjax API Overview



GlideAjax Methods

- `addParam()`
- `getXML()`
- `getXMLAnswer()`

Where Can I Use This?

Client Side

GlideForm



Client Scripts

GlideUser



UI Actions

GlideAjax



UI Scripts



Service Portal

Server Side

GlideRecord



Business Rules

GlideSystem



UI Actions

GlideDateTime



Script Includes



Scheduled Jobs



Service Portal



Web Services



Workflows

Section Recap

- Use GlideAjax when you need to access server-side data while on the client-side
- GlideAjax is available to you within any client-side scripting location
- Use Script Includes to store the server-side code you'd like to run with GlideAjax
- By default, XML is returned by the GlideAjax API
- Use APIs to encode/decode JSON

