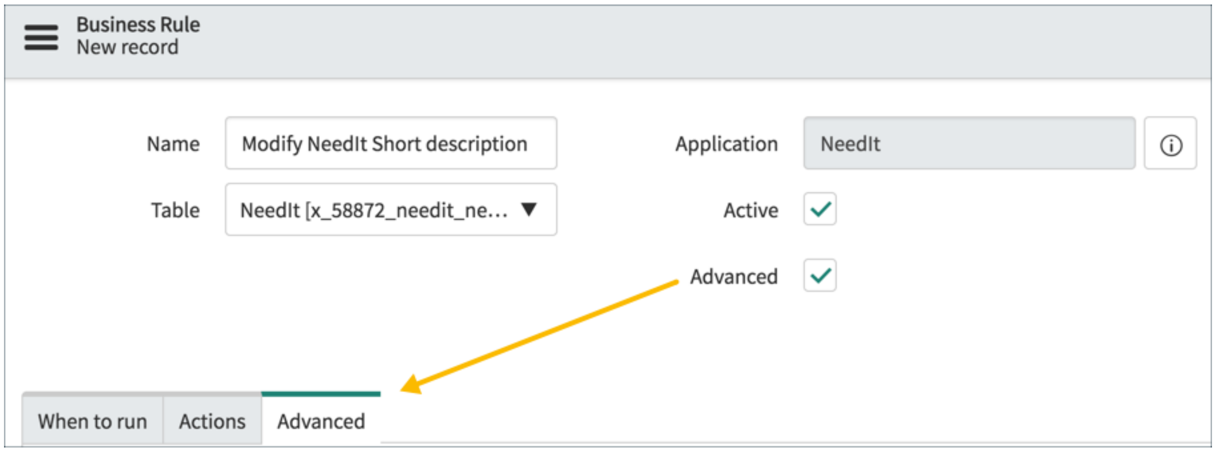
ServiceNow Application Developer

Server-side Scripting > Business Rule Scripts

Business Rules scripts use the server-side APIs to take actions. Those actions could be, but are not limited to:

- Invoking web services
- Changing field values
- Modifying date formats
- Generating events
- Writing log messages

The *Advanced* option must be selected to write Business Rule scripts. The scripting fields are in the *Advanced* section.



Business Rule
New record

Name	Modify NeedIt Short description	Application	NeedIt
Table	NeedIt [x_58872_needit_ne...	Active	✓
		Advanced	✓

There are two fields for scripting in the Advanced section:

- *Condition*
- *Script*

current and previous

Business Rules often use the *current* and *previous* objects in their script logic.

The *current* object is automatically instantiated from the *GlideRecord* class. The *current* object's properties are all the fields for a record and all the *GlideRecord* methods. The property values are the *values as they exist in the runtime environment*.

The *previous* object is also automatically instantiated from the *GlideRecord* class. The *previous* object's properties are also all fields from a record and the *GlideRecord* methods. The property values are the values for the record fields *when they were loaded from the database and before any changes were made*. The *previous* object is not available for use in async Business Rules.

The syntax for using the *current* or *previous* object in a script is:

```
<object_name>.<field_property>
```

An example script using *current* and *previous*:

```
// If the current value of the description field is the same as when the
// record was loaded from the database, stop executing the script
if(current.description == previous.description){
    return;
}
```

Condition Field

Use the *Condition* field to write Javascript to specify when the Business Rule script should execute. Using the *Condition* field rather than writing condition logic directly in the *Script* field avoids loading unnecessary script logic. The Business Rule script logic only executes when the *Condition* field returns *true*. If the *Condition* field is empty, the field returns *true*.

There is a special consideration for async Business Rules and the *Condition* field. Because async Business Rules are separated in time from the database operation which launched the Business Rule, there is a possibility of changes to the record between when the condition was tested and when the async Business Rule runs. To re-evaluate async Business Rule *conditions* before running, set the system property, *glide.businessrule.async_condition_check*, to *true*. You can find information about setting system properties on the **ServiceNow docs site (https://docs.servicenow.com)**.

The Condition script is an expression which returns *true* or *false*. If the expression evaluates to *true*, the Business Rule runs. If the condition evaluates to *false*, the Business Rule does not run.

This is CORRECT syntax for a condition script:

```
current.short_description == "Hello world"
```

This is INCORRECT syntax for a condition script:

```
if(current.short_description == "Hello world"){}
```

Some example condition scripts:

The value of the *State* field changed from anything else to 6:

```
current.state.changesTo(6)
```

The *Short description* field has a value:

```
!current.short_description.nil()
```

The value of the *Short description* field is different than when the record was loaded:

```
current.short_description != previous.short_description
```

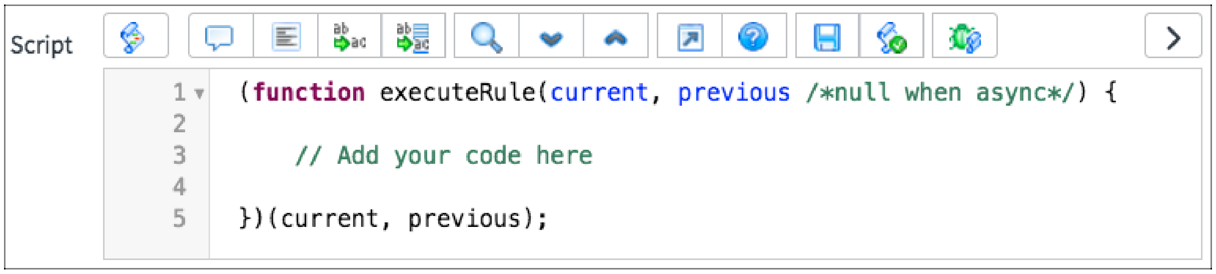
The examples use methods from the server-side API.

- The ***changesTo()* (https://developer.servicenow.com/app.do#!/api_doc?v=madrid&id=r_ScopedGlideElementChangesTo_Object_o)** method checks if a field value has changed from something else to a hardcoded value
- The ***nil()* (https://developer.servicenow.com/app.do#!/api_doc?v=madrid&id=r_ScopedGlideElementNil)** method checks if a field value is either *NULL* or the *empty string*

Notice that condition logic is a single JavaScript statement and does not require a semicolon at the end of the statement.

Script Field

The *Script* field is pre-populated with a template:



```
(function executeRule(current, previous /*null when async*/) {

    // Add your code here

})(current, previous);
```

Developers write their code inside the *executeRule* function. The *current* and *previous* objects are automatically passed to the *executeRule* function.

Notice the template syntax. This type of function syntax is known in JavaScript as a self-invoking function or an Immediately Invoked Function Expression (IIFE). This type of function is immediately invoked after it is defined. ServiceNow manages the function and when it is invoked.