



- 1 A snippet is a small template for code that performs a specific task in a specific language and language version.
- 2 Each snippet belongs to a specific topic. Examples may include: Promises, jQuery AJAX request, SQL Select statement, etc.
- 3 Each snippet is written for a specific programming language
- 4 Each snippet is written for a specific version of said programming language
- 5 Snippets may be accompanied by fully written, working code samples that demonstrate the topic. A snippet may have more than one related code sample.
- 6 Snippets may be accompanied by an explanation of the topic. There is only one explanation allowed for each snippet.
- 7 Snippet authors and consumers can tag the snippet with additional keywords to enhance search
- 8 Sometimes there is more than one way to perform a task. In this case, a snippet can have more than one variation. All related snippets would be stored in the Snippet table, but the relationship would be tracked in another table.
- 9 A snippet has one author. A code sample has one author. An explanation can have many authors (like a wiki).

CODE SAMPLE		
Field Name	Data Type	Key
ID	INT	PK
CodeSample	TEXT	
Snippet_ID	INT	FK

EXPLANATION		
Field Name	Data Type	Key
ID	INT	PK
Explanation	TEXT	
Snippet_ID	INT	FK

SNIPPET		
Field Name	Data Type	Key
ID	INT	PK
Title	TEXT	
Snippet	TEXT	
ShortDescription	TEXT	
Language_Version_ID	INT	FK
Topic_ID	INT	FK

SNIPPET_TAGS		
Field Name	Data Type	Key
Snippet_ID	INT	FK
Tag_ID	INT	FK

LANGUAGE		
Field Name	Data Type	Key
ID	Int	PK
Name	Text	
Version	Text	

TAGS		
Field Name	Data Type	Key
ID	Int	PK
Tag	Text	

TOPIC		
Field Name	Data Type	Key
ID	Int	PK
Name	Text	

Users		
Field Name	Data Type	Key
ID	INT	PK
Username	TEXT	
Password	TEXT	





http://

>_

A Big Title

[Home](#) | [About](#) | [Search](#) | [Add a Snippet](#)

Creating a Promise in JavaScript (ES6)

Snippet

```
//Declaration

let myFirstPromise = new Promise((resolve, reject) => {
});

//Use:

myFirstPromise.then((successMessage) => {
});
```

Explanation

[← Edit this Example](#)

A promise can be:

fulfilled - The action relating to the promise succeeded
rejected - The action relating to the promise failed
pending - Hasn't fulfilled or rejected yet
settled - Has fulfilled or rejected

The promise constructor takes one argument, a callback with two parameters, resolve and reject. Do something within the callback, perhaps async, then call resolve if everything worked, otherwise call reject.

Like throw in plain old JavaScript, it's customary, but not required, to reject with an Error object. The benefit of Error objects is they capture a stack trace, making debugging tools more helpful.

Examples

[← Add an Example](#)[One](#) [Two](#) [Three](#) [Four](#)

```
let myFirstPromise = new Promise((resolve, reject) => {
  var x = 5;
  var y = 10;
  resolve(x * y);
});
myFirstPromise.then((success) => {
  console.log(success); // console logs 50
});
```

Tags

[JavaScript Promises](#) [ES6](#) [Asynchronous](#) [Programming](#)

Rate this Code Sample

A Web Page

http://

>_

A Big Title

[Home](#) | [About](#) | [Search](#) | [Add a Snippet](#)

Q

search

Add a Snippet

Title

Short Description

Language

ComboBox

Snippet

Tags

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

☐ not selected

☒ selected

New Tag

Add Snippet

Ideas for Tag Submission:
Autocomplete?
Have Preset Tags Available,
Choose using Drop-down list or
Checkboxes?
Only add Tag if it doesn't already
exist?



http://

>_

A Big Title

[Home](#) | [About](#) | [Search](#) | [Add a Snippet](#)

Add Explanation

Snippet Title

//Declaration

```
let myFirstPromise = new Promise((resolve, reject) => {  
  });
```

//Use:

```
myFirstPromise.then((successMessage) => {  
  });
```



http://



>_

A Big Title

[Home](#) | [About](#) | [Search](#) | [Add a Snippet](#)

Add Code Sample

Snippet Title

//Declaration

```
let myFirstPromise = new Promise((resolve, reject) => {  
  });
```

//Use:

```
myFirstPromise.then((successMessage) => {  
  });
```



http://



Q Promises

Snippet	Topic	Language	Tags
Creating Promises using ES6	Promises	JavaScript (ES6)	#Asynchronous #JavaScript #Promises
Basic JavaScript Promises	Promises	JavaScript	



http://

>_

A Big Title

[Home](#) | [About](#) | [Search](#) | [Add a Snippet](#)

.....

Recent Snippets

Snippet	Topic	Language
Creating Promises using ES6	Promises	JavaScript (ES6)
Basic JavaScript Promises	Promises	JavaScript
Creating Promises using ES6	Promises	JavaScript (ES6)
Basic JavaScript Promises	Promises	JavaScript
Creating Promises using ES6	Promises	JavaScript (ES6)
Basic JavaScript Promises	Promises	JavaScript
Creating Promises using ES6	Promises	JavaScript (ES6)
Basic JavaScript Promises	Promises	JavaScript

Popular Snippets

Snippet	Topic	Language
Creating Promises using ES6	Promises	JavaScript (ES6)
Basic JavaScript Promises	Promises	JavaScript

Featured Snippet



JavaScript

Featured Snippet Title

A paragraph of **text** with an [unassigned link](#).
A *second row* of ~~text~~ with a [web link](#)

```
//Declaration

let myFirstPromise = new Promise((resolve, reject) => {
});

//Use:

myFirstPromise.then((successMessage) => {
});
```

A promise can be:

fulfilled - The action relating to the promise succeeded
rejected - The action relating to the promise failed
pending - Hasn't fulfilled or rejected yet
settled - Has fulfilled or rejected

The promise constructor takes one argument, a callback with two parameters, resolve and reject. Do something within the callback, perhaps async, then call resolve if everything worked, otherwise call reject.

Like throw in plain old JavaScript, it's customary, but not required, to reject with an Error object. The benefit of Error objects is they capture a stack trace, making debugging tools more helpful.