

# jQuery och Ajax

*Asynchronous JavaScript and XML*

# Agenda

- *Vad är jQuery?*
- *Varför jQuery?*
- *Hur använder vi jQuery?*
- *Vad är Ajax?*
- *jQuery och Ajax*

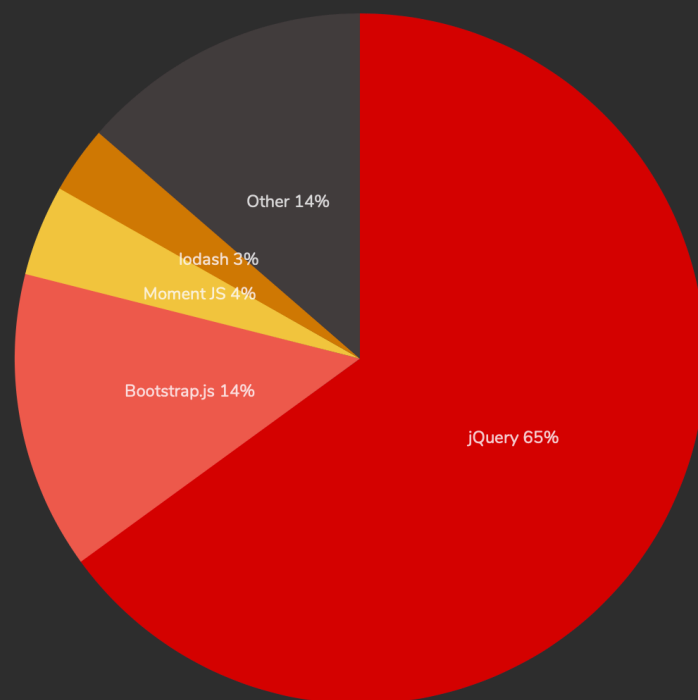
# Vad är jQuery?

- *Släpptes 2006 av John Resig*
- *Bibliotek (inte ett ramverk)*
- *Skapades för att underlätta arbete med DOM*
- *Stort ekosystem (både plugins och community)*

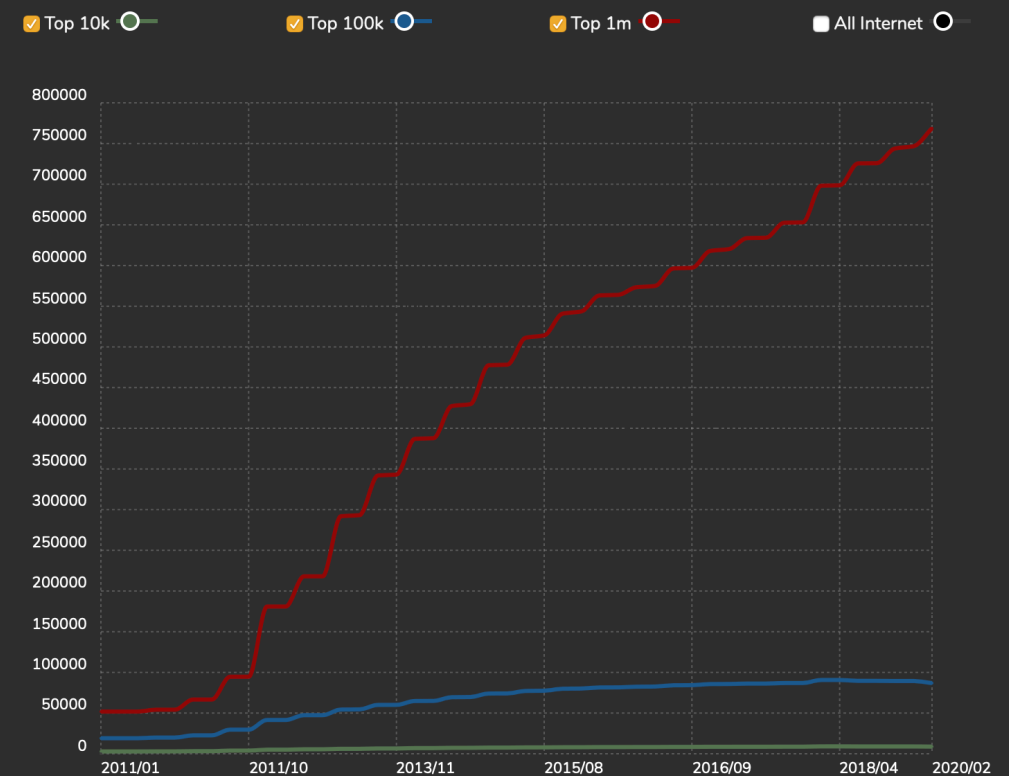
# Varför jQuery?

## JavaScript Library Usage Distribution in the Top 1 Million Sites

Statistics for websites using JavaScript Library technologies



## jQuery Usage Statistics



<https://trends.builtwith.com/javascript/javascript-library>

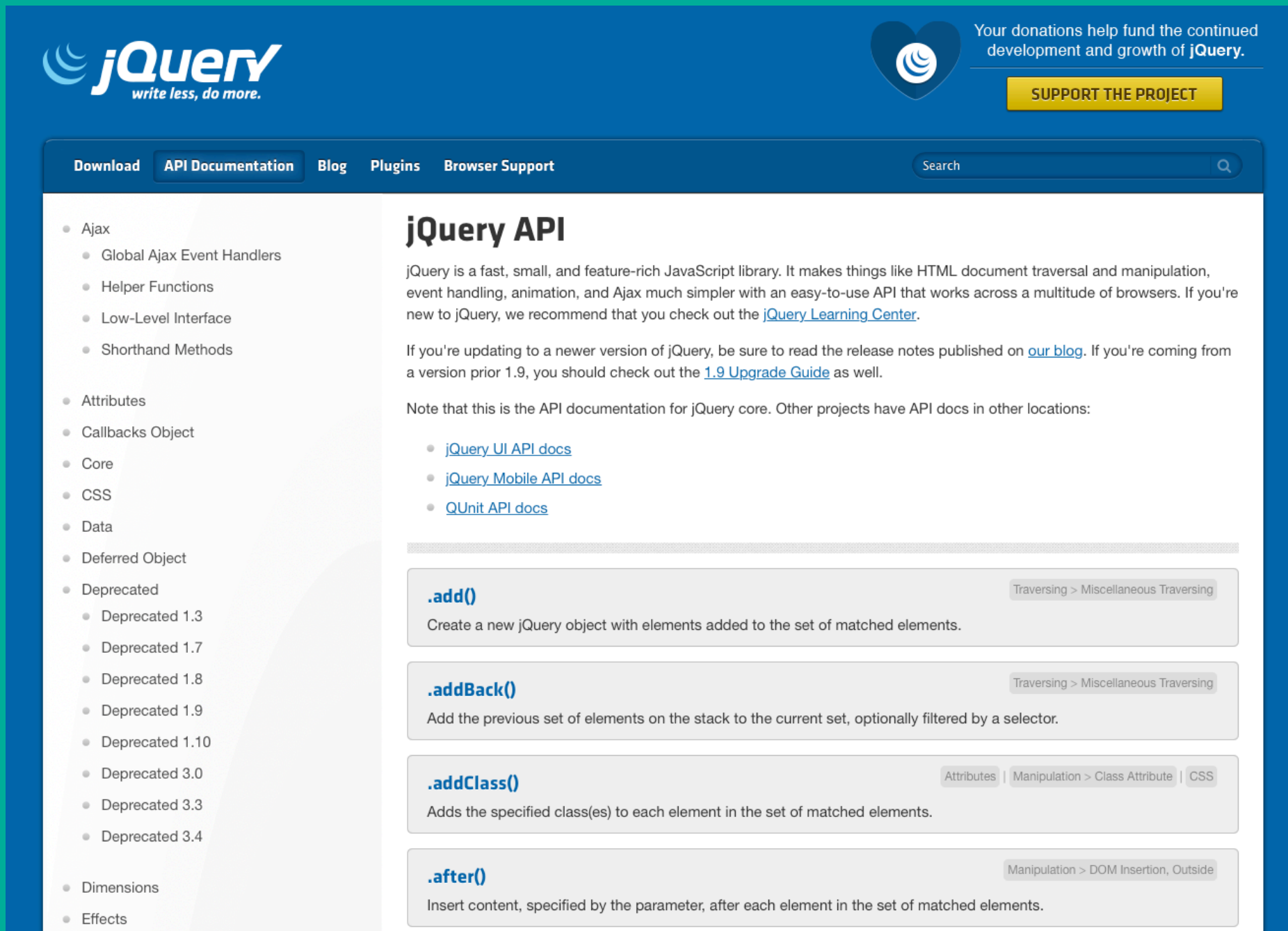
# Varför jQuery?

- *Underlättar arbete med DOM*
- *Hjälpfunktioner*
- *Cross-platform*
- *Plugins (bildspel, filuppladdning, m.m.)*
- *Community*
- *När ska man inte använda jQuery?*

# Tutorials

- *<https://www.codecademy.com/learn/learn-jquery>*
- *<https://learn.jquery.com/>*
- *<http://www.w3schools.com/jquery/>*
- *<https://www.codeschool.com/courses/try-jquery>*
- *<http://jqfundamentals.com/chapter/jquery-basics>*

# Dokumentation



The screenshot shows the jQuery API documentation website. At the top, there's a blue header with the jQuery logo and the tagline "write less, do more.". To the right, there's a heart icon and text stating "Your donations help fund the continued development and growth of jQuery." with a yellow button labeled "SUPPORT THE PROJECT". Below the header, a navigation bar contains links for "Download", "API Documentation" (which is highlighted), "Blog", "Plugins", and "Browser Support". A search bar is also present on the right side of the navigation bar. The main content area is divided into two columns. The left column features a sidebar with a list of categories: Ajax, Attributes, Callbacks Object, Core, CSS, Data, Deferred Object, Deprecated, Dimensions, and Effects. The "Deprecated" category is expanded, showing a list of deprecated versions from 1.3 to 3.4. The right column is titled "jQuery API" and contains an introductory paragraph about jQuery, a note about updating to newer versions, and a list of links to other API documentation (jQuery UI, jQuery Mobile, and QUnit). Below this, there are four method cards: ".add()", ".addBack()", ".addClass()", and ".after()", each with a brief description and a breadcrumb trail.

**jQuery API**

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. If you're new to jQuery, we recommend that you check out the [jQuery Learning Center](#).

If you're updating to a newer version of jQuery, be sure to read the release notes published on [our blog](#). If you're coming from a version prior 1.9, you should check out the [1.9 Upgrade Guide](#) as well.

Note that this is the API documentation for jQuery core. Other projects have API docs in other locations:

- [jQuery UI API docs](#)
- [jQuery Mobile API docs](#)
- [QUnit API docs](#)

**.add()** Traversing > Miscellaneous Traversing  
Create a new jQuery object with elements added to the set of matched elements.

**.addBack()** Traversing > Miscellaneous Traversing  
Add the previous set of elements on the stack to the current set, optionally filtered by a selector.

**.addClass()** Attributes | Manipulation > Class Attribute | CSS  
Adds the specified class(es) to each element in the set of matched elements.

**.after()** Manipulation > DOM Insertion, Outside  
Insert content, specified by the parameter, after each element in the set of matched elements.

<https://api.jquery.com>

# \$ = jQuery

- *En referens till jQuery sparas under variabelnamnet \$*



# jQuery

## jQuery()

Return a collection of matched elements either found in the DOM based on passed argument(s) or created by passing an HTML string.

```
1
2 // Hämta alla länkar
3 let links = $("a");
4
5 // Hämta alla element med klassen "blue"
6 let blueClass = $(".blue");
7
8 // Hämta elementet med id "start"
9 let start = $("#start");
10
11 // Hämta alla länkar
12 let links = jQuery("a");
13
14 // Hämta alla element med klassen "blue"
15 let blueClass = jQuery(".blue");
16
17 // Hämta elementet med id "start"
18 let start = jQuery("#start");
19
```

# jQuery vs vanilla

```
1
2 // Hämta alla länkar
3 let links = $("a");
4 let links = document.querySelectorAll("a");
5
6 // Hämta alla element med klassen "blue"
7 let blueClass = $(".blue");
8 let blueClass = document.querySelectorAll(".blue");
9
10 // Hämta elementet med id "start"
11 let start = $("#start");
12 let start = document.getElementById("start");
13
```

# jQuery

```
1
2 // Hämta elementet med id "start"
3 let start = $("#start");
4 // Dölj elementet
5 start.hide();
6 // Visa elementet
7 start.fadeIn();
8 // Lägg till klassen "center"
9 start.addClass("center");
10 // Ta bort elementet
11 start.remove();
12
```

# jQuery

```
1
2 // $(this) är detsamma som respektive <a>
3 $("a").each(function() {
4     let href = $(this).attr("href");
5     console.log(href);
6 });
7
8 // Dåligt: Lägg till en klass på alla paragrafer
9 $("p").each(function() {
10     $(this).addClass("center");
11 });
12
13 // Bättre: Lägg till en klass på alla paragrafer
14 $("p").addClass("center");
15
16 // Alla paragrafer blir klickbara, en alert-ruta
17 // öppnas med respektive paragrafs text
18 $("p").click(function() {
19     let txt = $(this).text();
20     window.alert(txt);
21 });
22
```

# jQuery

```
1
2 // Navigera DOM
3 // =====
4 // Hämta det nästkommande elementet från ett element
5 $("#my-element").next();
6 // Hämta det föregående elementet från ett element
7 $("#my-element").prev();
8 // Hämta föräldern till ett element
9 $("#my-element").parent();
10 // Hämta alla barnen av ett element
11 $("#my-element").children();
12 // Hitta ett element inom ett annat
13 $("#my-element").find(".blue");
14
```

# Exempel: jQuery

- *HTML5 attributet data-\*, hämta/ange data-värden*
- *Exempel 1: flikar*
- *Exempel 2: filtrera*
- *Exempel 3: modal*

# HTML5-attributet: data

- *Ibland räcker inte de attribut som finns att tillgå*
- *Attributet **data-\*** ger oss möjlighet att ange egna attribut med vilket värde vi själva vill*

```
1
2 <ul>
3   <li data-animal-type="bird">Owl</li>
4   <li data-animal-type="fish">Salmon</li>
5   <li data-animal-type="spider">Tarantula</li>
6 </ul>
7
```

# jQuery och data-\*

```
1
2 // Hämta värdet för attributet "data-id" på ett element
3 $("#my-element").attr("data-id");
4 // Ange värdet för attributet "data-id" på ett element
5 $("#my-element").attr("data-id", "new value");
6
7 // Vi kan göra något snarlikt för formulär
8 // Hämta värdet från (t.ex.) en <input>
9 $("#my-element").val();
10 // Ange värdet på en <input>
11 $("#my-element").val("new value");
12
```



# Exempel 1

- *Flikar*

# Exempel 2

- *Filtrera*

# Exempel 3

- *Modal (popup)*

# Vad är Ajax?

- *Kom '99 (Internet Explorer 5)*
- *Samling av webbt tekniker för att skapa interaktivitet*
- *Asynkront (ingen sidladdning, körs i bakgrunden)*
- *Används på de flesta webbplatser idag*

# Vilka webbt tekniker?

- *XMLHttpRequest*, ett objekt som tillåter en webbplats att göra anrop utan att sidan laddas om. Numera används `fetch`
- *Document Object Model*, modifiera befintligt innehåll
- *HTML* och *CSS*, märk upp och beskriv utseendet
- *XML* (vanligtvis *JSON*), format av en data som hämtas och skickas

# Användningsområden

- *Autocomplete (t.ex. när du söker på Google)*
- *Flerstegs betalningsformulär (t.ex. när du handlar online)*
- *Realtidssystem (t.ex. en live-chat, röstningssystem, m.m.)*
- *Skriva och skicka email med (t.ex.) Gmail*
- *Med mera.*

# För- och nackdelar

- *Kan göra en webbplats snabbare (t.ex. bättre prestanda)*
- *Kan göra en webbplats mer interaktiv*
- *Kan göra att en webbplats känns som en applikation*
- *Bakåtknappen kan sluta fungera*
- *Otillgänglig för de som har JS inaktiverat*
- *Kan ibland påverka indexering från sökmotorer*

# jQuery och Ajax

- *Idag finns även det inbyggda alternativet `fetch`*
- *[https://developer.mozilla.org/en-US/docs/Web/API/Fetch\\_API/Using\\_Fetch](https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch)*

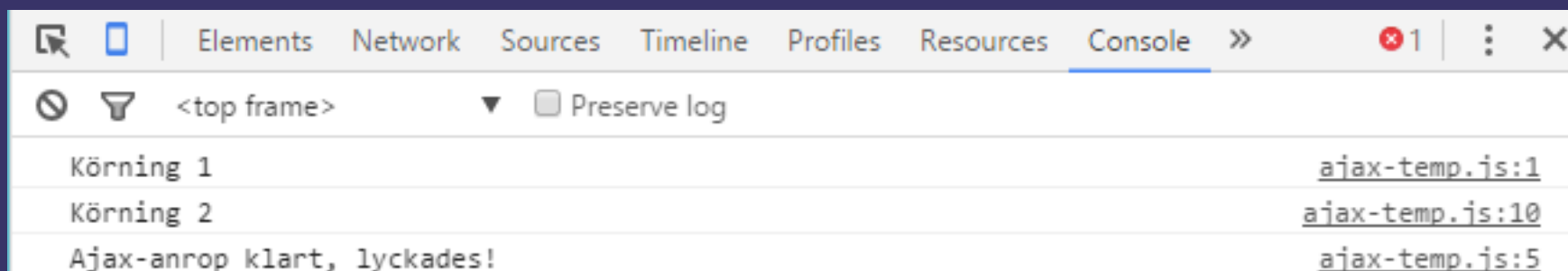


# jQuery och Ajax

```
1
2 console.log("Körning 1");
3
4 $.ajax({ url: "/" })
5   .done(function(data) {
6     console.log("Ajax-anrop klart, lyckades!");
7   })
8   .fail(function(data) {
9     console.log("Ajax-anrop klart, misslyckades!");
10  });
11
12 console.log("Körning 2");
13
```

# jQuery och Ajax

```
1
2 console.log("Körning 1");
3
4 $.ajax({ url: "/" })
5   .done(function(data) {
6     console.log("Ajax-anrop klart, lyckades!");
7   })
8   .fail(function(data) {
9     console.log("Ajax-anrop klart, misslyckades!");
10  });
11
12 console.log("Körning 2");
13
```



# jQuery och Ajax

```
1
2 console.log("Körning 1");
3
4 function onSuccess(data) {
5     console.log("Ajax-anrop klart, lyckades!");
6 }
7
8 function onFail(data) {
9     console.log("Ajax-anrop klart, misslyckades!");
10 }
11
12 $.ajax({ url: "/" })
13     .done(onSuccess)
14     .fail(onFail);
15
16 console.log("Körning 2");
17
```

# jQuery och Ajax

```
1
2 console.log("Körning 1");
3
4 $.ajax({
5     url: "/",           // URL
6     type: "GET",        // Metod
7     dataType: "JSON",   // Dataformat
8     data: {             // Parametrar (valfritt)
9         arg1: "value 1",
10        arg2: "value 2",
11        arg3: "value 3"
12    }
13 }).done(function(data) {
14     console.log("Ajax-anrop klart, lyckades!");
15 }).fail(function(data) {
16     console.log("Ajax-anrop klart, misslyckades!");
17 });
18
19 console.log("Körning 2");
20
```

# JSON

- *JavaScript Object Notation*

```
1
2 let jsonData = {
3   employees: [
4     { firstname: "John", lastname: "Doe" },
5     { firstname: "Jane", lastname: "Smith" }
6   ]
7 }
8
9 console.log(jsonData);
10 console.log(jsonData.employees);
11 console.log(jsonData.employees[1].lastname); // ?
12
13 // JavaScript objekt till sträng (JSON)
14 console.log(JSON.stringify(jsonData));
15 // JSON (sträng) till ett JavaScript objekt
16 console.log(JSON.parse("{}"));
17
```

# Frågor?