1

Basic Technologies

- Hypertext Markup Language
- Cascading Style Sheets

2

Interactive Web

- JavaScript
- Critical Rendering Path
- Json & Ajax
- jQuery
- React

JSON

JSON

- JavaScript Object Notation (JSON)
  Data format that represents data as a set of JavaScript objects
- natively supported by all modern browsers
  and libraries to support it in old ones
- not yet as popular as XML, but steadily rising due to its simplicity
  and ease of use

JSON

JSON

```
{
    "private":   "true",
    "from":      "Alice Smith (alice@example.com)",
    "to": [
            "Robert Jones (roberto@example.com)",
            "Charles Dodd (cdodd@example.com)"
    ],
    "subject":   "Tomorrow's event!",
    "message": {
            "language":  "english",
            "text":      "Hey guys, don't forget me!"
    }
}
```

JSON

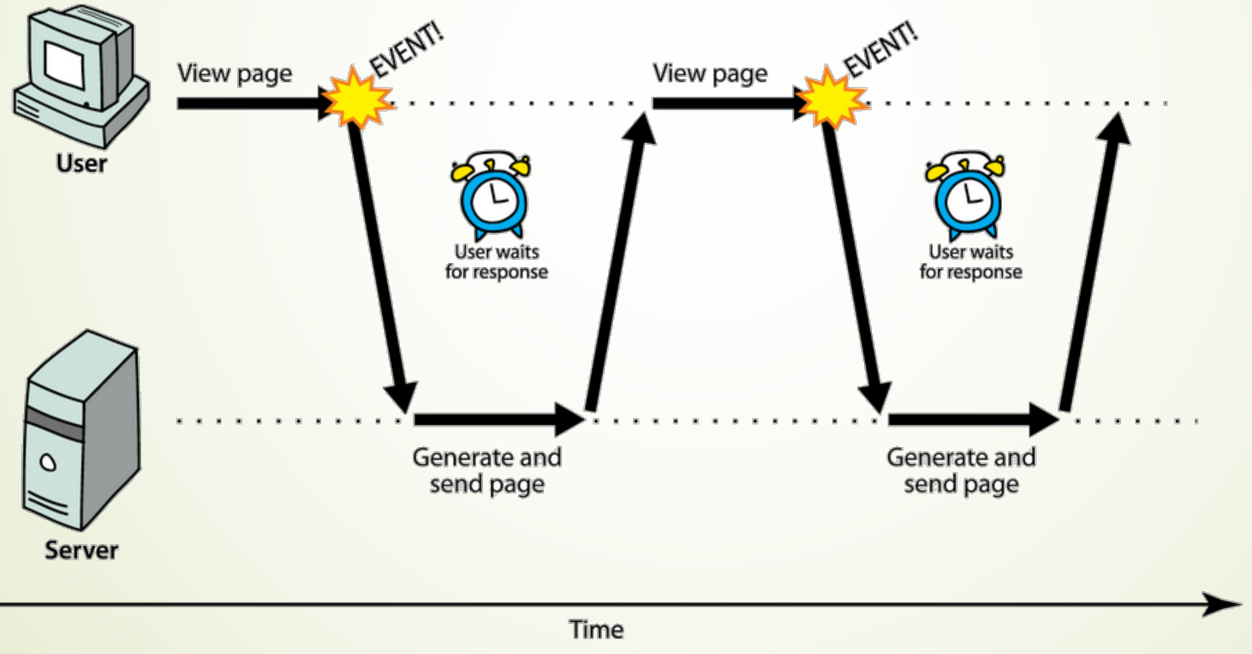
Browser JSON methods

- `JSON.parse(string)`
  converts the given string of JSON data into an equivalent JavaScript object and returns it
- `JSON.stringify(object)`
  converts the given object into a string of JSON data (the opposite of JSON.parse)

Web Communication

Synchronous

- pattern: click, wait, refresh

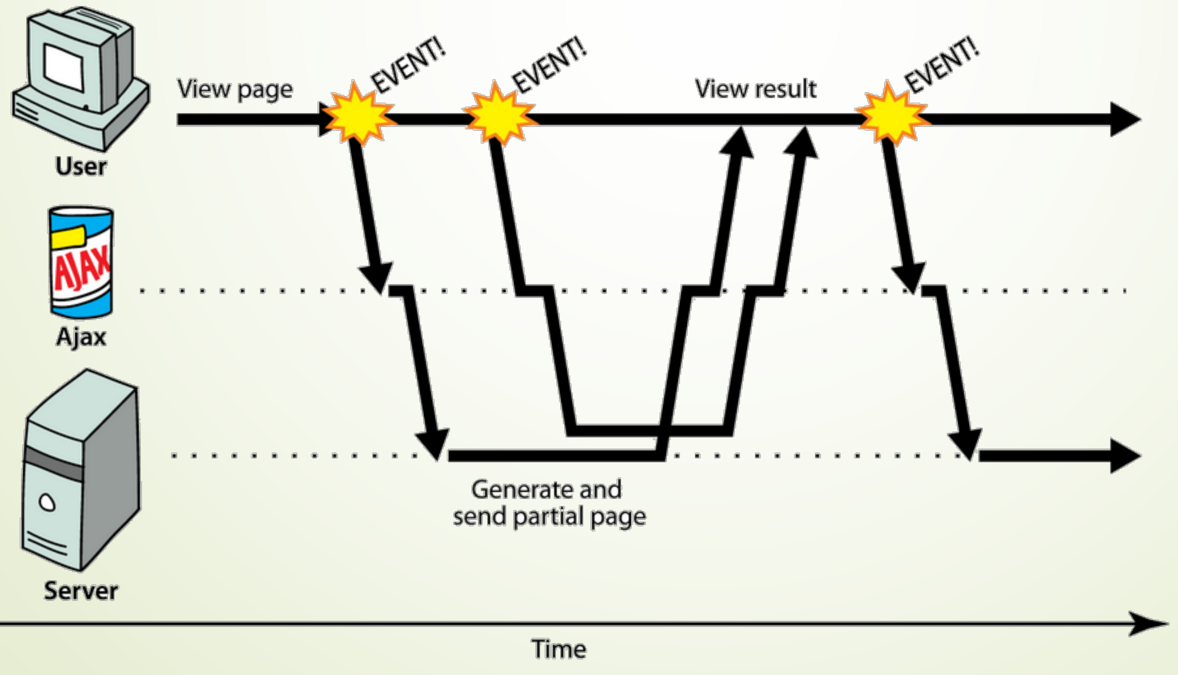


II.2

Web Communication

Asynchronous

- keep interacting with page while data loads
- communication pattern made possible by Asynchronous JavaScript and XML (AJAX)



AJAX

Technologies

- XHTML and CSS for presenting information
- DOM for dynamically interacting with and displaying the information presented
- XMLHttpRequest object to manipulate data asynchronously with the Web server
  - update a web page without reloading the page
  - request data from a server - after the page has loaded
  - receive data from a server - after the page has loaded
  - send data to a server - in the background
- XML, HTML, and XSLT for data interchange and manipulation
- JavaScript for binding data requests and information display

AJAX

XMLHttpRequest

- prepare data
- determine processing
- send request

```
var xrq = new XMLHttpRequest();

xrq.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        document.getElementById("demo").innerHTML =
            this.responseText;
    }
};
xrq.open("GET", "ajax_info.txt", true);
xrq.send();
```

- the response is available as a string or as a parsed XML document in the `responseText` and `responseXML` properties
- use JavaScript to use the response and update the current page's DOM

1

Basic Technologies

- Hypertext Markup Language
- Cascading Style Sheets

2

Interactive Web

- JavaScript
- Critical Rendering Path
- Json & Ajax
- jQuery
- React

jQuery

jQuery

- powerful javascript library
  - many functions to make programming tasks easier
  - uses short-hand notations
- jQuery is accessible as a global function and as an object instance
  - function “jQuery”, abbreviated as “$”
- provides additional utility functions
  - frequent pattern: `$.fname(parameters)`
- supports event handlers
  - frequent pattern: `DOMObject.eventname(function)`
  - convenient pattern: using local anonymous functions

jQuery

jQuery - Selectors

- syntax is tailor-made for selecting HTML elements and performing some action on the element(s)
- basic syntax: `$(selector).action()`
  - a $ sign to define/access jQuery
  - a (selector) to "query (or find)" HTML elements
  - a jQuery action() to be performed on the element(s)

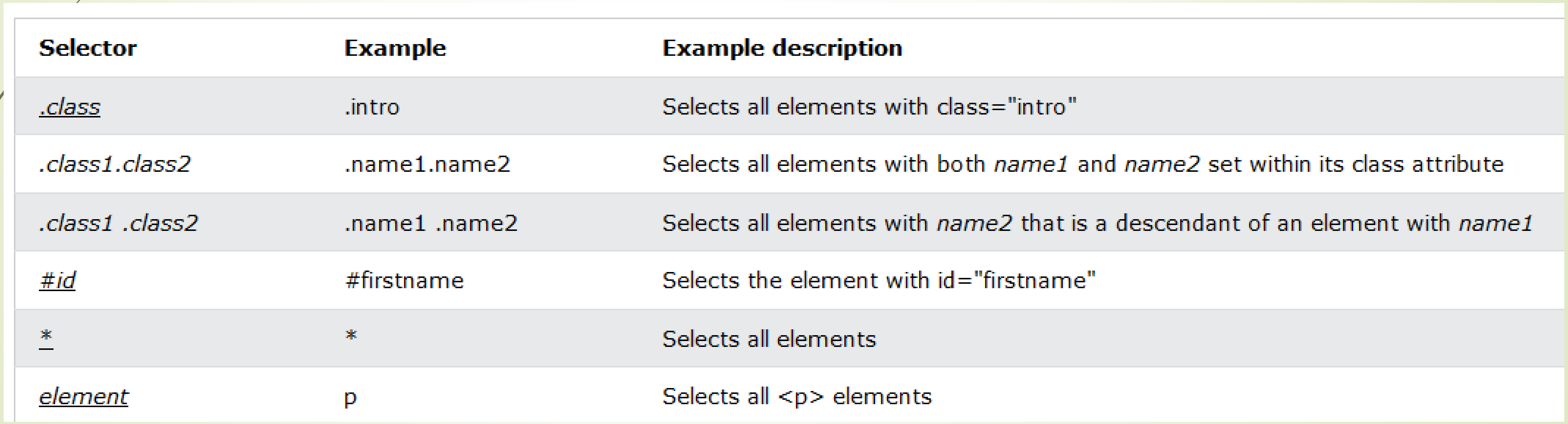
```
$(this).hide()        hides the current element
$("p").hide()         hides all <p> elements
$(".test").hide()     hides all elements with class="test"
$("#test").hide()     hides the element with id="test"
```

jQuery

JavaScript / jQuery - Selectors

```
document.querySelector(CSS selectors)
```

```
$(CSS selectors).action()
```



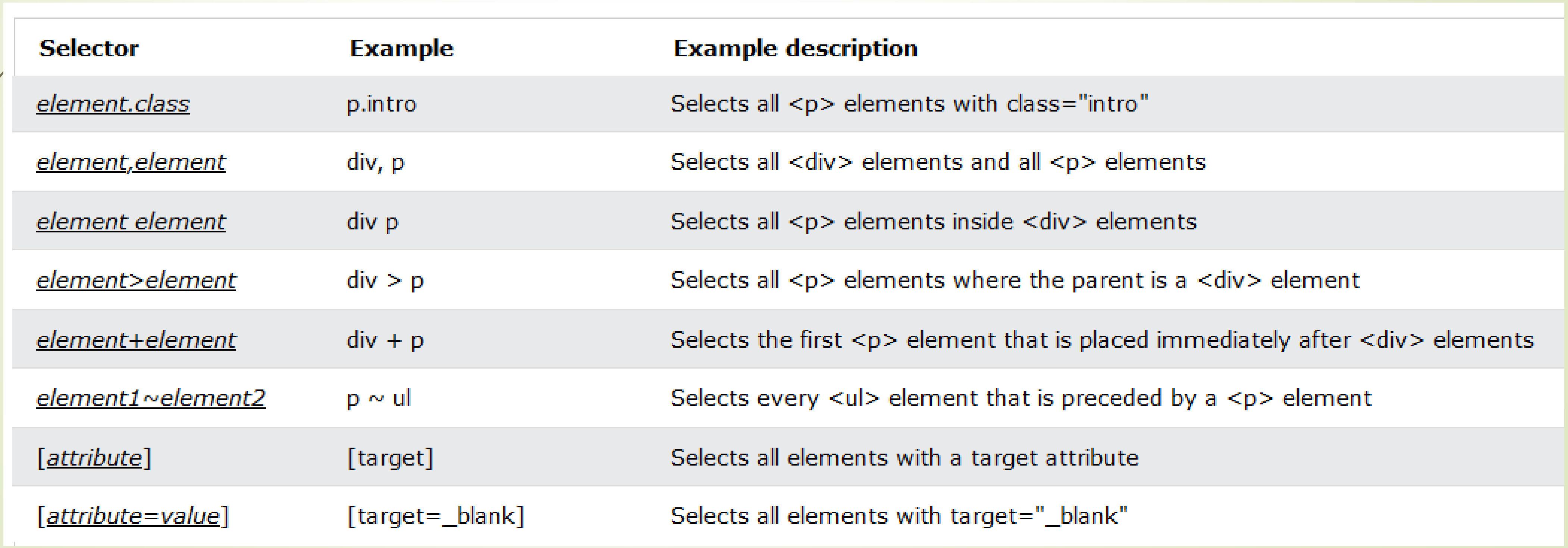
Selector	Example	Example description
.class	.intro	Selects all elements with class="intro"
.class1.class2	.name1.name2	Selects all elements with both *name1* and *name2* set within its class attribute
.class1 .class2	.name1 .name2	Selects all elements with *name2* that is a descendant of an element with *name1*
#id	#firstname	Selects the element with id="firstname"
*	*	Selects all elements
element	p	Selects all <p> elements

jQuery

JavaScript / jQuery - Selectors

```
document.querySelector(CSS selectors)
```

```
$(CSS selectors).action()
```



Selector	Example	Example description
element.class	p.intro	Selects all <p> elements with class="intro"
element,element	div, p	Selects all <div> elements and all <p> elements
element element	div p	Selects all <p> elements inside <div> elements
element>element	div > p	Selects all <p> elements where the parent is a <div> element
element+element	div + p	Selects the first <p> element that is placed immediately after <div> elements
element1~element2	p ~ ul	Selects every <ul> element that is preceded by a <p> element
[*attribute*]	[target]	Selects all elements with a target attribute
[*attribute=value*]	[target=_blank]	Selects all elements with target="_blank"

jQuery

jQuery – Example

```
$('div').each( function(index, value) {
  console.log('div${index}: ${this.id}');
});
```

jQuery

jQuery – Callback functions

- JavaScript statements are executed line by line. However, with effects, the next line of code can be run even though the effect is not finished. This can create errors.
- a callback function is executed after the current effect is finished.
- typical syntax: `$(selector).hide(speed, callback);`

```
$("button").click( function() {
  $("p").hide("slow", function() {
    alert("The paragraph is now hidden");
  });
});
```

jQuery

jQuery – AJAX

```
$('#btn').click(function() {
    var selIdsText = $('#mysongs input:checked').map(function() {
        return $(this).parents('tr').children().first();
    }).text();
    $.ajax({
        type: 'POST',
        url: 'serverDummy.php',
        data: {selection: selIdsText},
        success: function(data) {
            ...
        }
    });
});
```

II.2

jQuery

Javascript versus jQuery

Javascript Code	jQuery Code

`window.onload = onDocumentReady;`
`function onDocumentReady() {`
`    // body of function`
`}`
	`$(document).ready(function(){`
	`  // body of function`
	`});`

`document.querySelector("nav ul")`	`$("nav ul");`
`document.querySelector("h1").innerHTML = "Welcome";`	`$("h1").text("Welcome");`

`var button = document.querySelector("button");`
`button.onclick = function() {`
`  // body of function`
`}`
	`$("button").click(function() { ...});`

`document.querySelector("h1").style.color = "red";`	`$("h1").css("color", "red")`