DAI: Mòdul 10
jQuery

Xavier Noguero
Carles Mateu http://carlesm.com
Ciències de la Computació i Intel·ligència Artificial
Universitat de Lleida

Llibreries JavaScript

Motivacions tècniques:

- Hi ha molts tipus de navegadors, donant lloc a moltes incompatibilitats amb el codi JavaScript (cross-browser).
- Millorar la interactivitat amb l'usuari pot ser molt dur utilitzant únicament JavaScript.
- Manipulació del DOM.
- Events
- Suport per a Ajax.
- Crear efectes visuals, animacions, widgets, etc.

Llibreries JavaScript

Motivacions personals:

- No cal reinventar la roda
- Fer més feina amb menys codi
- Estalviar temps
- Es pot ser un bon programador, però n'hi ha de millors
- Velocitat
- El codi resultant s'entén més

Llibreries de JavaScript

- **Mootools**: "El framework javascript compacte"
  - Senzill, ben planificat.
  - http://mootools.net/
- **JQuery**: "Llibrería Javascript per a escriure menys i fer més"
  - És una de les que té més acceptació.
  - Molt documentada.
  - Codi net i elegant.
  - http://jquery.com/
- **Dojo**: "Grans experiències... per a cadascú"
  - Atractiu i molt estès.
  - http://www.dojotoolkit.org/

Llibreries de JavaScript

- **Prototype**: "El framework javascript amb el propòsit de facilitar el desenvolupament d'aplicacions dinàmiques"
  - Força estès.
  - Una mica pesat
  - http://www.prototypejs.org/
- **Script.aculo.us**
  - Basat en Prototype
  - http://script.aculo.us/
- **YUI**: "The Yahoo! User Interface Library"
  - Creat per Yahoo!
  - Una mica pesat.
  - http://developer.yahoo.com/yui/

Llibreries de JavaScript

- **GWT Google Web Toolkit**: "Construeix aplicacions Ajax amb java"
  - Desevolupat en Java per Google.
  - Disposa d'un compilador que converteix el codi java en Javascript i html compatible per a tots els navegadors.
  - http://code.google.com/webtoolkit/
- Altres:
  - **Qooxdoo**: http://qooxdoo.org/
  - **Mochikit: http://mochikit.com/**
  - **Rico**: http://openrico.org/rico/home.page
  - **Ext JS**: http://extjs.com/

Quina escollim?

- L'elecció ha de respondre satisfactòriament a:
  - Fa el que necessitem per al nostre projecte?
  - Com de ràpid és el codi?
    - Vegeu els tests de http://mootools.net/slickspeed/
  - Quina mida ocupa la llibrería?
    - Moltes es poden obtenir també en javascript comprimit.
  - La documentació és suficient i adequada?
  - És activa la comunitat que hi ha darrera?
  - És modular el codi?
  - Qui utilitza la llibrería? Exemples:
    - Digg i Apple utilitzen Prototype/Script.aculo.us, cNet i GameSpot utiltizen MooTools, WordPress utilltiza jQuery...
  - Quina permet estalviar més temps programant?

jQuery

- Creada per en John Resig al 2006
- Codi concís, lleuger i extensible.
- Versió actual: **1.5.1**
- Què aporta? en què es centra?
  - Manipulació del DOM.
  - Events i animacions.
  - Ajax.
  - Plugins: per a extendre la llibrería amb noves funcionalitats.
- Altres característiques:
  - 29KB (versió comprimida).
  - Navegadors: IE6.0+, FF2.0+, Safari 3.0+, Opera 9.0+, Chrome.
  - Suport per a CSS3.

jQuery

- Com començar?
  - Descarregar la llibrería: http://www.jquery.org
  - Incloure el fitxer **jquery.js**:

```
<script src="jquery/jquery.js" type="text/javascript"></script>
```

  - Suposem que l'hem descarregat dins una carpeta "jquery" a l'arrel de l'aplicació.

- També la podem carregar directament d'Internet:

```
<script src="http://code.jquery.com/jquery-latest.js"></script>
```

Primers pasos

Una operació fonamental amb jQuery, és seleccionar una part concreta d'un document.

- Utilitzem $(*selector*) o bé jQuery(*selector*) selector pot contenir qualsevol selector CSS.
- Exemple:

```
$('span').addClass('cursiva');
```

  - $() implementa un iterador implícit (bucle for).
  - Amb .addClass() “injectem” la classe CSS (l'hem de definir al nostre fitxer .css) 'cursiva' a tots els elements 'span' del document.

Primers pasos

- Exemples d'ús de $() per seleccionar elements:
  - Un nom d'etiqueta html:

```
$('p')          //obtenim tots els paràgrafs
```

  - Un id:

```
$('#un-id')     //obtenim l'element del document que té per id="un-id"
```

  - Una classe:

```
$('.una-classe') //obtenim tots els elements que tenen per classe "una-classe"
```

- Nota: si useu altres llibreries JS que també utilitzen $(), feu servir jQuey() per a evitar conflictes!

```
jQuery.noConflict();
```

Primers pasos

- Suposem una funció JavaScript qualsevol:

```
function fesQuelcom(){
	$('span').addClass('cursiva');
}
```

- Abans, per a que s'executés en carregar la pàgina:

```
<body onload=″fesQuelcom()″>
```

- Ara, amb jQuery:

```
$(document).ready(fesQuelcom);

//o bé, de manera equivalent, sense necessitat de definir la funció (funcions anònimes):

$(document).ready(function(){
	$('span').addClass('cursiva');
});
```

Selectors amb jQuery

Primers pasos – Selectors

- Podem utilitzar selectors:
  - CSS (de l'especificació 1 a la 3),
  - XPath (fins la versió 1.2),
  - o personalitzats.
- No cal preocupar-se per les incompatibilitats entre CSS i el navegador (especialment amb IE).
- XPath:
  - s'utilitza per a identificar elements o els seus valors en documents XML.
  - jQuery en suporta un conjunt bàsic de selectors.

Ús de selectors CSS amb jQuery

- Accedint als fills d'un element:

```
$('#id > div').addClass('una-classe');
```

  - Només seran fills els que pengen directament de l'element amb id “id”.
- Si aquests fills tenen més elements div dins seu, per trobar-los podem fer:

```
$('#id div:not(.una-classe)').addClass('una-altra-classe');
```

  - Així accedim a tots els div que conté l'element amb id “id” i que no tenen aplicada la classe “una-classe”.

Ús de selectors XPATH amb jQuery 1.2

- Per a seleccionar elements que tenen un atribut concret:

```
$('a[@title]').addClass('una-classe');
```

- Per a seleccionar un element contingut dins un altre, no fem servir l'@:

```
$('div[span]').addClass('una-classe');
```

- Jquery ha eliminat del Core aquests selectors Xpath a favor d'incorporar els corresponents de CSS.
- Aquests encara estan disponibles mitjançant un plugin (http://plugins.jquery.com/project/xpath/)

Reemplacem XPATH per CSS a la v1.3

- Per trobar els elements que tenen un atribut:

```
$('a[title]').addClass('una-classe');
```

  - Amb la versió 1.3, per a seleccionar un atribut només fem servir [] i sense usar l'@.
- Ex: com aplicar estils diferents segons el tipus d'hipervincle:

```
$('a[href^=mailto:]').addClass('enllaç-email');        //amb ^ indiquem que “comença per...”

$('a[href$=.odp]').addClass('enllaç-impress'); //amb $ indiquem que “acaba amb...”

$('a[href^=http] [href*=udl]').addClass('enllaç-udl');      //amb * indiquem que “conté...”
```

Jquery també té selectors personalitzats

- Així accedim al segon div del document que tingui aplicada la classe “una-classe”:

```
$('div.una-classe:eq(1)').addClass('una-altra-classe');
```

  - Fixeu-vos que comença a comptar a partir de 0 com els arrays de JavaScript.
- Cerca per text:

```
$('p:contains(“universitat”)').addClass('una-altra-classe');
```

  - Apliquem la classe “una-altra-classe” a tots els paràgrafs del document que contenen la paraula “universitat”.

Selectors de formularis

- Per a cercar elements d'un formulari:
  - :text, :checkbox, :radio, :image, :submit, :reset, :password, :file => cerca els elements “input” amb el valor del selector a l'atribut “type”
  - :input => Input, textarea, select i elements button.
  - :button => botons
  - :enabled => elements del formulari “enabled”
  - :disabled => per a elements desabilitats
  - :checked => botons radio o checkbox marcats
  - :selected => elements d'opció seleccionats

- Convé no abusar d'aquest recurs si volem que el codi sigui entenedor

Manipulant les files d'una taula

- Per millorar la llegibilitat de les files d'una taula podem alternar dos estils diferents a les files:

```
$('tr:odd').addClass('un-color');        //odd i even són-0 based, compte!
$('tr:even').addClass('un-altre-color');
```

- Si hi ha més d'una taula a la mateixa pàgina pot passar que no comencin totes pel mateix color. Això és pot solucionar amb selectors CSS3:

```
$('tr:nth-child(even)').addClass('un-color');   //nth-child és 1-based
$('tr:nth-child(odd)').addClass('un-altre-color');
```

Mètodes per navegar pel DOM

- Amb els selectors vistos podem cercar conjunts d'elements, però de vegades necessitem més flexibilitat.
- Aquests mètodes, sovint utilitzen paràmetres amb els mateixos noms que els selectors vistos.
  - Exemple:

```
$('tr:odd').addClass('un-color');

// es pot reescriure amb el mètode .filter() com:

$('tr').filter(':odd').addClass('un-color');
```

  - .filter() accepta una funció com a paràmetre per fer cerques més depurades:

```
$('tr').filter(function(index){
      return index==1 || $(this).attr("id") == "un-id-concret";
}).addClass('un-color');       // aplicarà l'estil al segon tr i al que tingui per id="un-id-concret"
```

Relacionant-se amb la família...

- Per accedir al pare d'un element:

```
$('li').parent().addClass('negreta');
```

- Per accedir al germà del costat:

```
$('li').next().addClass('blau');
```

- Per accedir a tots els germans següents:

```
$('li').nextAll().addClass('blau');
```

  - També existeixen .prev() i .prevAll()
- Per accedir a tots els altres germans:

```
$('li').siblings().addClass('blau');
```

Relacionant-se amb la família...

- Per incloure el mateix element amb els altres:

```
$('li').siblings().andSelf().addClass('blau');
```

- Una altra manera de seleccionar tots els germans:

```
$('li').parent().children().addClass('blau');
```

Encadenament d'accions. Si volem fer diferents accions a partir d'un mateix element, les podem encadenar en una única sentència:

```
$('li').parent().find('li:eq(1)').addClass('blau').end().find('td:eq(2)')
                                                    .addClass('vermell');
```

- També podem trencar-la en diferents línies per millorar la llegibilitat del codi.

Accedint als elements DOM

- Totes les expressions de selectors i la majoria de mètodes jQuery, retornen un objecte jQuery.
- Si volem obtenir un objecte DOM directament, el mètode get() cobreix aquest propòsit:

```
var etiqueta = $('#ident').get(0).tagName;
```

  - Així podem obtenir el tagname d'un element.
  - Fixeu-vos que get() accepta un index numèric.
- Una abreviació de get() es pot escriure així:

```
var etiqueta = $('#ident')[0].tagName;
```

  - És una sintaxi similar als arrays de JavaScript.

Events amb jQuery

Events

- bind(): permet especificar un event de JS i assignar-li un comportament. Exemple:

```
$(document).ready(function(){
   $('span').bind('click', function() {
       alert('has fet click');
   });
});
```

  - Poden coexistir múltiples crides a .bind(), afegint tants comportaments adicionals al mateix event com sigui necessari.
  - S'executaran en el mateix ordre en que s'hagin registrat.

Context d'event - this

- Quan declarem un comportament d'un event, podem accedir a l'element on s'aplica amb this:
  - $(this) serà un objecte jQuery que representarà l'element DOM.
  - this, serà l'element DOM directament.
- Exemple:

```
$(document).ready(function(){
  $('div').bind('click', function() {
      if(this.id == 'contenidor'){
            $(this).removeClass().addClass('una-classe');
  });
});
```

  - Segons necessitem utilitzar mètodes jQuery o propietats natives de JS, utilitzarem un o altre.

Events

- unbind(): permet desvincular un comportament d'un event JS Exemple:

```
//quan fem clic a un enllaç, s'executarà la funció fesQuelcom:
$("a").bind("click", fesQuelcom);

//quan volguem desvincular l'execució de la funció fesQuelcom quan fem clic a un enllaç:
$("a").unbind("click", fesQuelcom);
```

- A més:

```
$("a").unbind("click");
```

  - Desvinculem tots els comportaments asignats en fer clic a un enllaç

```
$("a").unbind();
```

  - Desvinculem tots els events asignats a un enllaç

Events – mètodes abreviats

- Per reduir l'escriptura de codi, disposem de mètodes abreviats per a tots els events reconeguts de JS:
  - blur
  - change
  - click
  - dblclick
  - error
  - focus
  - keydown
  - keypress
  - keyup
  - load
  - mousedown
  - mousemove
  - mouseout
  - mouseover
  - mouseup
  - resize
  - scroll
  - select
  - submit
  - unload

```
$(document).ready(function(){
	$(“a”).click(fesQuelcom);
	$(“#contenidor”).resize(function() {
		$(this).addClass('iluminat');
	});
});
```

Altres mètodes d'events - toggle()

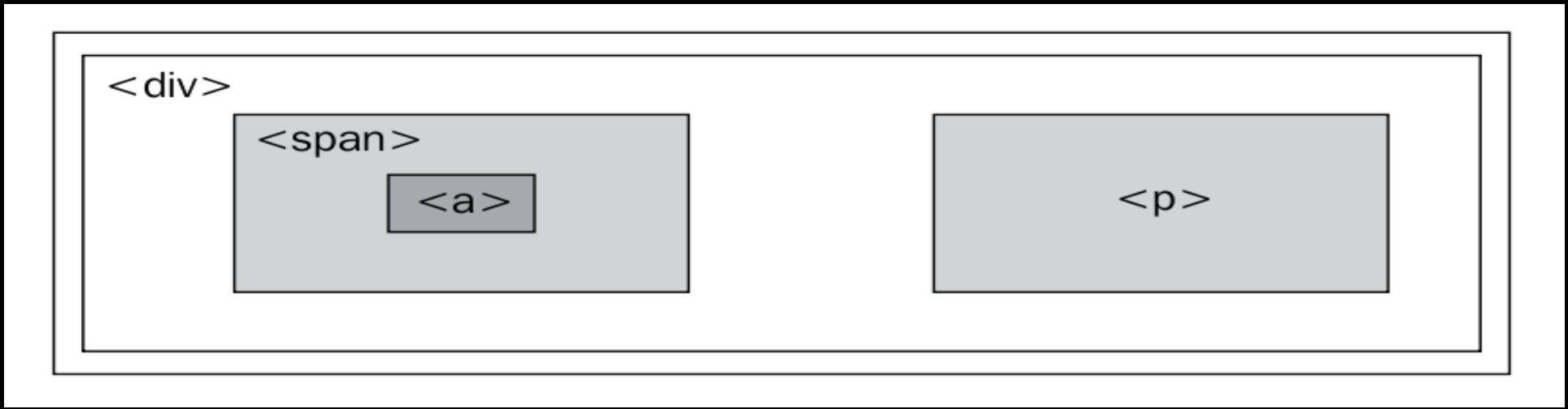
- toggle(): permet alternar diferents comportaments quan es produeix un clic sobre un element DOM.
- Definim aquests comportaments en funcions:

```
$(document).ready(function(){
	$('#contenidor').toggle(function(){
		$('p').removeClass().addClass('vermell');
	}, function() {
		$('p').removeClass().addClass('blau');
	});
});
```

  - Quan fem un clic sobre l'element #contenidor, els paràgrafs seran vermells, i quan tornem a fer clic, seran de color blau.
  - Podem dues o més funcions.

El viatge d'un event

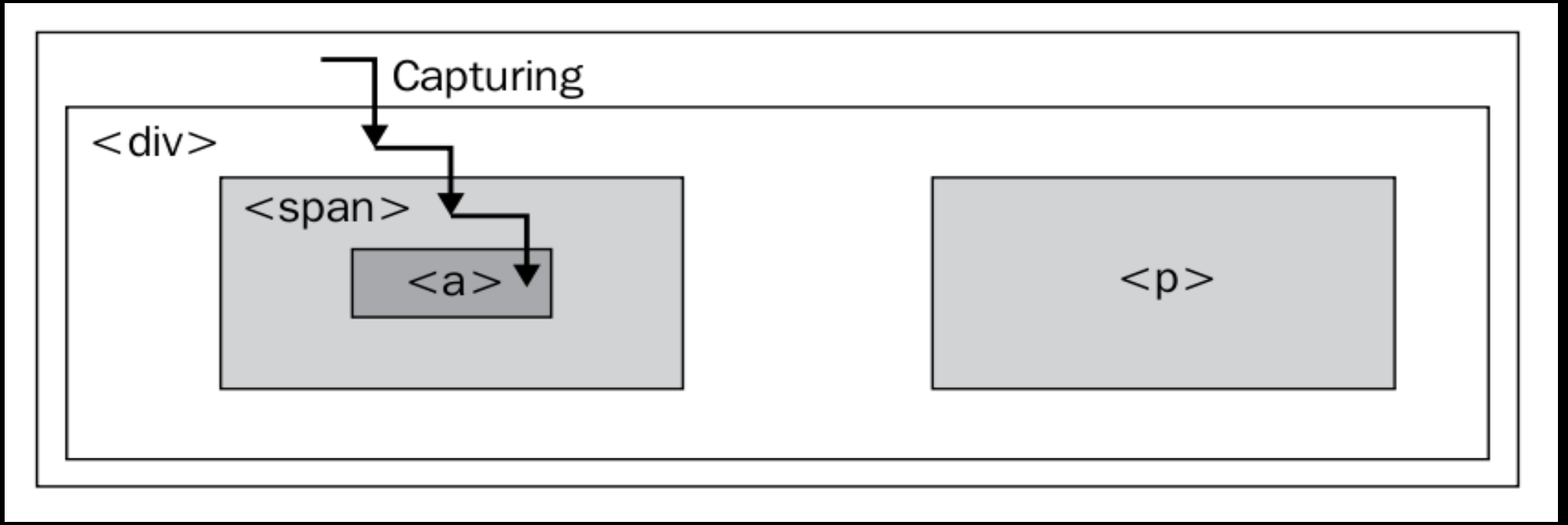
- Per a qualsevol event, poden haver-hi múltiples elements que poden reaccionar al mateix.
- Per exemple, donada una estructura com aquesta:



  - Quan fem clic sobre l'enllaç, , i <a> tenen l'oportunitat de respondre al clic, perquè tots tres es troben sota el cursor del ratolí al mateix temps.

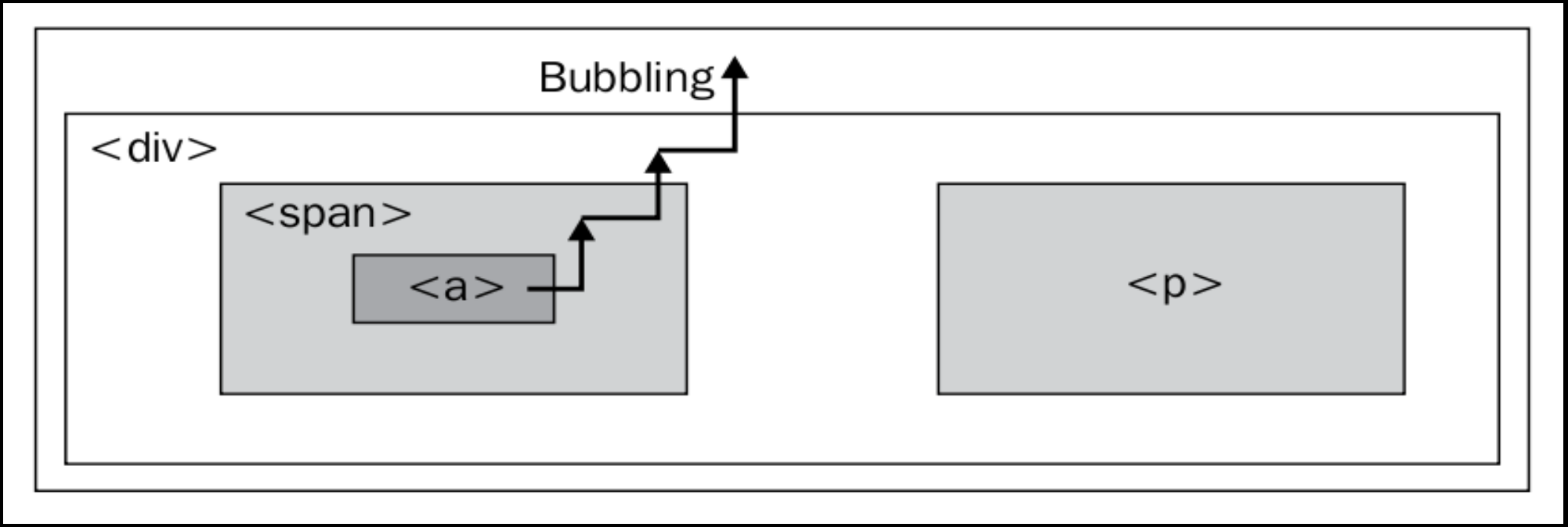
Estratègies de propagació d'events

- Per a permetre que diferents elements puguin reaccionar davant d'un event, hi han dues estratègies:
  - event capturing: l'event es rep a l'element més general i es propaga cap al més específic:



Estratègies de propagació d'events

- Per a permetre que diferents elements puguin reaccionar davant d'un event, hi han dues estratègies:
  - event bubbling: l'event es rep a l'element més específic i es propaga cap al més general:



Què fan els navegadors? I jQuery?

- Sorprenentment, diferents desenvolupadors van adoptar diferents models de propagació per als navegadors.
- L'estàndard DOM diu que totes dues estratègies haurien de poder ser utilitzades, primer d'elements generals a específics, i després l'event hauria de retornar propagant-se cap a amunt dins l'arbre DOM.
- jQuery segueix el model de bubbling, i podem assumir que un event sempre el rebrà l'element més específic i s'anirà propagant cap al més general.

Efectes no desitjats de bubbling - hover()

- Events com mouseover o mouseout poden generar efectes no desitjats amb la propagació d'events.
- hover(): permet alternar diferents comportaments segons si s'està per sobre d'un element DOM o no.
- També definim els comportaments en funcions:

```
$(document).ready(function(){
    $('#contenidor .boto').hover(function(){
        $(this).addClass('vermell');
    }, function() {
        $(this).removeClass('vermell');
    });
});
```

Quan passem per sobre d'un element que té la classe “boto” i que es troba dins l'element amb id “contenidor”, se li aplica la classe “vermell”, i quan sortim de sobre, es restaura com estava.

Alterant el viatge d'un event

- Hi ha ocasions en què hover no serveix. Suposem el cas que vulguem que quan es faci clic sobre un div que conté botons, aquests s'amaguin.
- El problema apareix quan fem clic a un botó, ja que també s'amagaran i això no ho desitgem.
- Per sol·lucionar-ho, hem d'accedir a l'objecte event i a la propietat target, que indica el primer element del DOM que ha rebut l'event:

```
$(document).ready(function(){
	$('#contenidor').click(function(event){
		if(event.target == this){
			$('#contenidor .button').toggleClass('amaga');
		}
	});	//així ens assegurem que l'ítem clicat serà sobre el div, i no sobre qualsevol
});		// dels seus sub-elements.
```

Aturant la propagació d'events

- L'objecte event disposa del mètode .stopPropagation(), que finalitza el procés de bubbling per aquell event.
- Alternativament, podem solucionar el problema anterior actuant sobre el comportament dels botons quan s'hi faci clic:

```
$(document).ready(function(){
    $('#contenidor' .button).click(function(event){
        //fés quelcom quan es faci clic al botó...
        event.stopPropagation();
    });    //
```

  - Així ens assegurem que l'event no arribarà mai al div contenidor quan es faci clic a un botó i només respondrà a l'event el botó.

Efectes i animacions amb jQuery

Efectes

- Abans d'entrar de ple amb efectes...
- Veurem el mètode .css():

```
var mida = $('p').css('font-size');              //obtenim el valor d'una propietat

$('p').css('font-size', '12');                   //modifiquem una propietat

$('p').css({'font-size':'12', 'color':'red'}); //modifiquem més d'una propietat alhora
```

  - Permet obtenir/inicialitzar propietats d'estil
- Nota: si es tracta d'aplicar un estil deduïble a priori, millor definir-lo al fitxer d'estils i fer servir .addClass() / .removeClass()!!

Efectes visuals

- .show(), .hide(), .toggle()
- Funcionalitat: mostrar o ocultar un element.
- Amb .toggle() aconseguim les dues funcions, un clic es mostra, un altre clic s'oculta l'element.
- Exemples:

```
$('p').hide();              // Oculta un element
$('p').show();              // Mostra un element
$('p').toggle('fast');          // Si estava ocult, es mostra i si es veia, s'oculta
$('p').show('slow');    // Mostra l'element amb una animació suau
$('p').hide(850);           // Oculta l'element amb una animació personalitzada (sense comes)
$('p').show('slow', fesQuelcomDespres)  // Executarà una funció després de l'efecte
```

- La velocitat és modulable: 'slow', 'normal' o 'fast'.
- També accepta un número en mil·lisegons.

Efectes visuals

- .fadeIn(), .fadeOut(), .fadeTo()
- Funcionalitat: mostrar o ocultar gradualment un element modulant-ne l'opacitat.
- Velocitat: 'slow', 'normal', 'fast' o un número.
- Tots tres mètodes accepten funció de retorn.
- Exemples:

```
$('p').fadeOut('slow');         // Desapareix el paràgraf lentament
$('p').fadeIn('nomal', fesQuelcom);   // Apareix el paràgraf i es crida la funció fesQuelcom

$('p').fadeTo('fast', 0.50);   // Modifica l'opacitat que tenia el paràgraf
                               // fins al 50% de visibilitat
```

Efectes visuals

- .slideDown(), .slideUp(), .slideToggle()
- Funcionalitat: mostra cap a baix o oculta cap a dalt un element ajustant l'alçada de l'element.
- Velocitat: 'slow', 'normal', 'fast' o un número.
- Tots tres mètodes accepten funció de retorn.
- Exemples:

```
$('p').slideDown(500);        // Desapareix el paràgraf
$('p').slideUp('fast');       // Apareix el paràgraf

$('p').slideToggle('normal'); // Si es veia, s'oculta, i si no es mostra
```

Efectes visuals

- .animation()
- Funcionalitat: crear animacions personalitzades.
- Paràmetres:
  - representaran l'estat final desitjat de l'element o elements a animar.
  - només propietats d'estil que admetin valors numèrics (Noms: border-color -> borderColor)
- Opcionals: velocitat, tipus, i/o funció de retorn.
- Exemple:

```
$('p').animation({left: '+=45'}, ,500);
```

Efectes visuals

- Podem combinar els mètodes introduïts per a aconseguir efectes composts:

```
$('p')
  .fadeTo('fast',0.5)
  .animate({
    'left': 35, borderWidth:'+=5'
  }, {duration: 'slow', queue: false})
  .fadeTo('slow',1.0)
  .slideUp('slow')
  .queue(function () {
        $(this).addClass("verd");
        $(this).dequeue();
   }
  .slideDown('slow');
```

Manipulant el DOM amb jQuery

DOM – Atributs d'elements

- Per veure, modificar i esborrar atributs d'un element (id, rel, title...) presentem els mètodes:
  - .attr('atr') (retorna el valor de l'atribut atr)
  - .attr({'atr':'val','atr1':'val1',...})
  - .removeAttr(atr)
- Exemple:

```
$(document).ready(function(){
	$('div > p').each(function(index){		//amb .each() creem un iterador explícit
		$(this).attr('id','paragraf-' + index);	//on disposem d'un index que es va
	});						//autoincrementant a cada element
});
```

DOM – Inserció de nous elements a dins

- Inserció d'elements al final o a l'inici d'altres:
  - .append() / .appendTo()
  - .prepend() / .prependTo()
- Exemple:

```
$(document).ready(function(){
	$('<p>Hola!</p>').appendTo('#un-id');

		//o bé, de manera equivalent...

	$('#un-id').append('<p>Hola!</p>');
});
```

DOM – Inserció de nous elements a fora

- Inserció d'elements abans o després d'altres:
  - .insertBefore() / .before()
  - .insertAfter() / .after()
- Exemple:

```
$(document).ready(function(){
	$('<a href="www.udl.cat">UdL</a>').insertBefore('div');

		//o bé, de manera equivalent...

	$('div').before('<a href="www.udl.cat">UdL</a>');
});
```

DOM – Altres

- .html()
- .text()
- .wrap() / .wrapAll() / .wrapInner()
- .replaceWith() / .replaceAll()
- .empty() / .remove()
- .clone()
- Exemple:

```
$(document).ready(function(){

    $('p').text('Hola de nou!');
    $('p').wrap('<div></div>');
});
```

AJAX amb jQuery

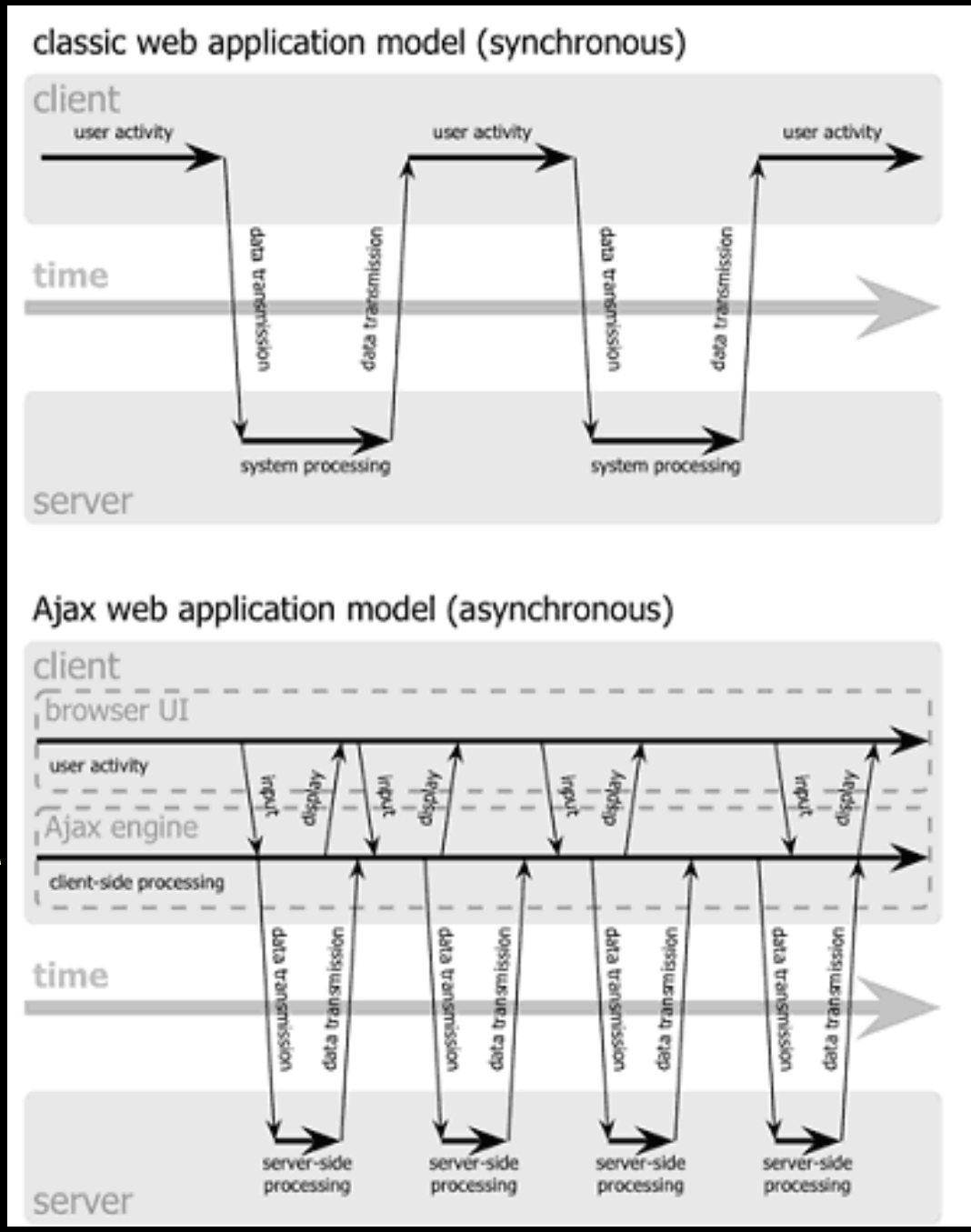
AJAX

- Sigles de Asyncronous JavaScript and XML
- Objectiu:
  - Permet carregar continguts dinàmicament sense necessitat de refrescar tota la pàgina
- És la suma de:
  - Peticions de tipus XMLHttpRequest (XHR) +
  - HTML, XML (o formats de dades similars), CSS, JavaScript
- Beneficis:
  - Interfícies d'usuari molt més interactives i atractives.
  - Millorar la usabilitat substancialment.

AJAX

Procés de càrrega de dades sota demanda:

- El navegador demana informació.
- En general, la petició és asíncrona, i l'usuari no nota interrupcions.
- El navegador rep la resposta.
- S'interpreta la informació rebuda i es mostra a l'usuari.



AJAX – JSON

- Els objectes JavaScript és poden definir com a parells clau-valor: {clau: valor}
- Els arrays es defineixen amb []
- Així un exemple d'objecte pot ser:

```
{
    "clau", "valor",
    "clau 2": {
        "array",
        "de",
        "valors"
    }
}
```

- JavaScript Object Notation (JSON) defineix dades utilitzant aquesta sintaxi senzilla.

AJAX – Segons el tipus de dades...

- Codi html: `$().load('fitxer.html');`
- Format Json: `$.getJson('fitxer.json');`
- Format JS: `$.getScript('fitxer.js');`
- Petició GET: `$.get('url');`
- Petició POST: `$.post('url');`
- Petició HTTP: `$.ajax('url');  // + configurable`
- Totes accepten (opcionalment) paràmetres de configuració i la definició d'una funció que s'executarà quan es rebi la informació solicitada (callback).

Plugins amb jQuery

Estenent noves funcionalitats

- Repositori disponible: http//plugins.jquery.com
- Com utilitzar-los?

  - Carreguem el codi font del plugin:

```
<script src="jquery.plugin.js" type="text/javascript"></script>
```

  - Per cridar-lo:

```
$('p').elMeuPluggin();
```

Alguns pluggins destacats

- Taules ordenables per columna:
  - http://tablesorter.com
- Aplicant estils a formularis:
  - http://www.dfc-e.com/metiers/multimedia/opensource/jqtransform
- Autocompletar controls de text de formularis:
  - http://bassistance.de/jquery-plugins/jquery-plugin-autocomplete/
- Validació de formularis:
  - http://jquery.bassistance.de/validate/demo/
- Degradats de fons i voreres arrodonides:
  - http://plugins.jquery.com/project/backgroundCanvas
- Un recull de plugins: http://www.kollermedia.at/archive/2007/11/21/the-ultimate-jquery-plugin-list/

jQuery User Interface

- És una suite plugins composada d'interaccions, widgets i efectes per a la interfície d'usuari.
- Per utilitzar-la, a més de carregar jQuery:

```
<script type="text/javascript" src="jquery-ui-1.7.1.custom.min.js"></script>
```

- Interaccions:
  - Draggable
  - Droppable
  - Resizable
  - Selectable
  - Sortable
- Widgets:
  - Accordion
  - Datepicker
  - Dialog
  - Progressbar
  - Slider
  - Tabs
- jQuery UI: http://jqueryui.com/
- ThemeRoller: http://jqueryui.com/themeroller/

Referencies

- jQuery Web Page:
  - http://jquery.com/
- jQuery User Interface:
  - http://jqueryui.com/
- JavaScript Library Overview:
  - http://www.slideshare.net/jeresig/javascript-library-overview
- Selectors CSS 3 - W3C:
  - http://www.w3.org/TR/css3-selectors/