IFT3225-H2016	Technologies de l'internet	
Examen Intra	Date:26 février 2016, 9:30-11:30	Guy Lapalme

AUCUNE DOCUMENTATION PERMISE

(10) 1. Soit l'URL suivant:

`http://www.iro.umontreal.ca/~lapalme/FIOTT/HTML/ch03s02.html#sec_complextype`

Faites ressortir les différentes composantes de cette adresse et expliquez chacune d'elles brièvement.

(10) 2. Décrivez ce qu'on appelle un *constructeur* en Javascript et comparez cette notion avec un constructeur en Java.

(10) 3. Expliquez ce qu'on entend par *capture* et *bouillonnement* des événements en Javascript. Comment peut-on les spécifier et donnez des exemples d'utilisation de chacun de ces concepts.

(30) 4. Soit le fichier d'instances XML suivant qui décrit une banque de CVs. Chaque `CV` comporte un identificateur unique (attribut `id`) et les liens entre les personnes sont identifiées par l'attribut `ref` de l'élément `connait` qui doivent référer à des `id` existant. Chaque CV peut comporter plusieurs `experience` et `competence`. Les dates de début et de fin d'expérience sont des dates en format ISO.

```
<CVs>
   <CV id="I0056" >
       <nom>Océrosse</nom> <prenom>Catherine</prenom>
       <experiences>
          <experience debut="2002-02-01" fin="2004-03-01">poinçonneuse</experience>
          <experience debut="2005-03-04" fin="2007-03-01">programmeuse</experience>
          <experience debut="2008-06-25" fin="2009-03-01">analyste</experience>
       </experiences>
       <competences>
          <competence>C++</competence>
          <competence>Java</competence>
          <competence>Pascal</competence>
       </competences>
       <connait refs="I0078 I0057"/>
   </CV>
   <CV id="I0057" >
       <prenom>Noémie</prenom> <nom>Nestrone</nom>
       <experiences>
          <experience debut="2002-02-01" fin="2004-03-01">Goûteuse d'eau</experience>
          <experience debut="2005-03-04" fin="2007-03-01">Laveuse de chats</experience>
       </experiences>
       <competences>
          <competence>Excel</competence>
          <competence>SAP</competence>
       </competences>
       <connait refs="I0078"/>
   </CV>
   <CV id="I0078" >
       <nom>de Grèce</nom> <prenom>Eustache</prenom>
       <experiences>
          <experience debut="2002-02-01" fin="2004-03-01">Dégorgeur d'escargots</experienc
       </experiences>
       <competences>
          <competence>Couteau</competence>
          <competence>Fourchette</competence>
          <competence>Cuillère</competence>
       </competences>
       <connait refs="I0057"/>
   </CV>
</CVs>
```

(15) (a) Donnez un schéma RelaxNG (forme compacte) qui permet de le valider (voir rappel de la syntaxe RelaxNG en annexe).

(15) (b) Donnez une feuille de style XSL (voir rappel de la syntaxe XSLT en annexe) qui permet de transformer ce fichier d'instance dans la page XHTML suivante, où les liens entre personnes sont indiquées par les noms plutôt que par les identificateurs.

3 curriculum vitae

Catherine Océrosse	
experiences	• poinçonneuse (25 mois) • programmeuse (24 mois) • analyste (8 mois)
competences	• C++ • Java • Pascal
connaît	Eustache de Grèce Noémie Nestrone

Noémie Nestrone	
experiences	• Goûteuse d'eau (25 mois) • Laveuse de chats (24 mois)
competences	• Excel • SAP
connaît	Eustache de Grèce

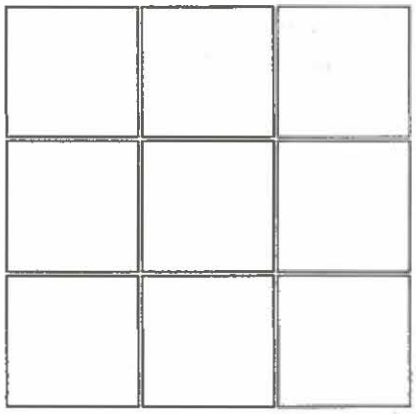
Eustache de Grèce	
experiences	• Dégorgeur d'escargots (25 mois)
competences	• Couteau • Fourchette • Cuillère
connaît	Noémie Nestrone

Pour calculer le nombre de mois entre deux dates `d1` et `d2`, utilisez l'expression Xpath suivante:

```
round(days-from-duration(xs:date($d1)-xs:date($d2)) div 30)
```

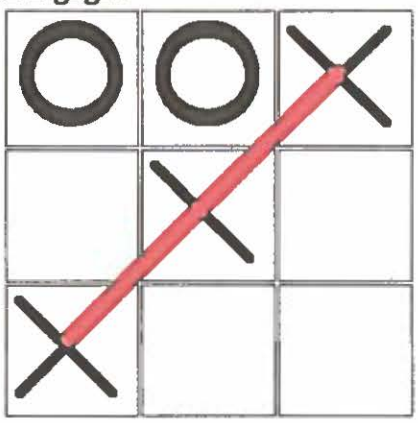
(40) 5. Vous devez programmer en Javascript une surface de jeu de Tic-Tac-Toe sur un tableau 3 par 3 (chaque case est de 100 pixels par 100 pixels) où les joueurs placent alternativement un X ou un O en cliquant dans une case du tableau. Une fois une case remplie, on ne peux changer le signe qui y a été mis. Le jeu arrête lorsqu'un des joueurs réussit à placer trois de ses signes dans une même ligne, colonne ou diagonale. On déclare une partie nulle quand les neuf cases sont remplies sans qu'aucune ligne, colonne ou diagonale de mêmes signes n'ait pu être obtenue par un des joueurs. Lorsque un joueur a gagné, on affiche une ligne rouge pour indiquer les cases *gagnantes*. Les deux figures suivantes montrent l'état initial et un état final du jeu.

Jouez à tic-tac-toe



Jouez à tic-tac-toe

X a gagné



Pour dessiner cette ligne, pensez à superposer un canvas sur le tableau de jeu que vous pouvez positionner de manière absolue.

Vous pouvez supposer qu'il y a dans le même répertoire que votre programme un fichier `X.gif` qui est l'image d'un X et `O.gif` qui est l'image d'un O.

Vous pouvez aussi utiliser (**sans la définir**, une version vous est fournie en annexe) la fonction `fini(jeu)` où `jeu` est un tableau 3 par 3 où chaque valeur du tableau est soit `""`, `"X"` ou `"O"` selon que la case est respectivement vide, contient un `X` ou un `O`. `fini` retourne une structure de la forme suivante:

```
{message:"...",ligne:[i1,j1,i2,j2]}
```

où message est une chaîne indiquant qui a gagné, avec les coordonnées de la ligne complétée (elle débute dans la case `i1,j1` et termine dans la case `i2,j2`). Pour une partie nulle, cette liste est vide. La fonction retourne `null` dans le cas où le jeu peut continuer.

Vous pouvez utiliser les fonctions du framework jQuery rappelé en annexe.

(10) (a) Donnez le code HTML correspondant à la page initiale (voir la figure de gauche)

(10) (b) Donnez le contenu de la feuille de style CSS pour ce jeu.

(20) (c) Donnez le code Javascript qui permet de jouer ce jeu.

Cet examen comporte 6 questions pour un total de 100 points.

Bonne chance

Rappels RelaxNG compact

id = *pattern*

Sortes de *patterns*

`element` *nom* { *pattern* }
`attribute` *nom* { *pattern* }
`mixed` { *pattern* }
2 *patterns* combinés avec , ou & ou |
pattern suivi de ? ou * ou +
id
`text`

Rappels HTML

Structure d'une page

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8"/>
  <link href="..." type="stylesheet"/>
  <title>...</title>
</head>
<body>
  ...
</body>
</html>
```

Éléments du body

`em,strong`	mise en relief
`sub,sup`	indice, exposant
`ul`	liste non-ordonnée
`ol`	liste ordonnée
`li`	élément de liste
`div,span`	regroupements d'éléments
`a href="..."`	lien entre documents
`img src="..." alt="..."`	image
`table`	tableau
`caption`	légende d'un tableau
`tr`	ligne d'un tableau
`th`	case d'entête de tableau
`td`	case de tableau

Éléments de division HTML5

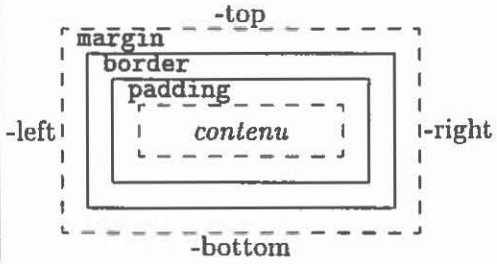
`header`	entête de toute la page
`nav`	navigation principale
`aside`	information auxiliaire
`section`	contenu de la page
`article`	unité d'information autonome
`footer`	pied de page

Rappels CSS

Sélecteurs

nom	nom d'un élément
.classe	nom d'une classe
#id	identificateur d'un élément
:pseudo-classe	`:link`, `:hover`, `:active`

Modèle de boîte



Principales propriétés

`background-color`	couleur du fond
`background-image`	`url(`*image*`)`
`background-repeat`	`repeat-x | repeat-y | no-repeat`
`background-position`	`left | top | center | bottom | right`
`border`	`border-width border-style` *couleur*
`color`	spécification de la couleur
`display`	`inline, block`
`font`	`font-family font-style font-weight`
`height`	hauteur du bloc
`margin`	largeur de la marge
`padding`	largeur du *padding*
`position`	`relative | absolute | fixed`
`text-align`	`left | right | center | justify`
`top`	position en haut à gauche
`width`	largeur du bloc

B.6. XSLT

Table B.6. Table 5.1, Section 5.1

<xsl:stylesheet> *xsl:import*,* *(declaration|xsl:variable|xsl:param)** **</xsl:stylesheet>**
<xsl:template match="pattern" name="QName"**>** *xsl:param*, sequence-constructor** **</xsl:template>**
<xsl:param name="QName" select="expression"**>** *sequence-constructor* **</xsl:param>**
<xsl:apply-templates select="expression"**>** *(xsl:sort*|xsl:with-param)** **</xsl:apply-templates>**
<xsl:call-template name="Qname"/>
<xsl:with-param name="QName" select="expression"**>** *sequence-constructor* **</xsl:with-param>**
<xsl:function name="QName"**>** *xsl:param*, sequence-constructor** **</xsl:function>**
<xsl:value-of select="expression"**>** *sequence-constructor* **</xsl:value-of>**
<xsl:variable name="QName" select="expression"**>** *sequence-constructor* **</xsl:variable>**
<xsl:copy> *sequence-constructor* **</xsl:copy>**
<xsl:copy-of select="expression"/>
<xsl:if test="expression"> *sequence-constructor* **</xsl:if>**
<choose> *xsl:when*, xsl:otherwise?* **</choose>**
<xsl:when test="expression"> *sequence-constructor* **</xsl:when>**
<xsl:otherwise> *sequence-constructor* **</xsl:otherwise>**
<xsl:for-each select="expression"> *xsl:sort*, sequence-constructor* **</xsl:for-each>**
<xsl:for-each-group select="expression" group-by="expression"**>** *xsl:sort*, sequence-constructor* **</xsl:for-each-group>**

```
<xsl:sort select="expression" data-type="{string}">
    sequence-constructor
</xsl:sort>
```

```
<xsl:element name="{string}">
    sequence-constructor
</xsl:element>
```

```
<xsl:text>
    character data
</xsl:text>
```

```
<xsl:attribute name="{string}" select="expression">
    sequence-constructor
</xsl:attribute>
```

```
<xsl:attribute-set name="QName" use-attribute-sets="Qnames">
    xsl:attribute*
</xsl:attribute-set>
```

```
<xsl:message>
    sequence-constructor
</xsl:message>
```

Selectors

Basics

- **#id**
- **element**
- **.class**
- **.class.class**
- *
- **selector1, selector2**

Hierarchy

- **ancestor descendant**
- **parent > child**
- **prev + next**
- **prev ~ siblings**

Basic Filters

- **:first**
- **:last**
- **:not(selector)**
- **:even**
- **:odd**
- **:eq(index)**
- **:gt(index)**
- **:lt(index)**
- **:header**
- **:animated**
- **:focus**

Content Filters

- **:contains(text)**
- **:has(selector)**
- **:parent**

Visibility Filters

- **:hidden**
- **:visible**

Child Filters

- **:nth-child(expr)**
- **:first-child**
- **:last-child**
- **:only-child**

Attribute Filters

- **[attribute]**
- **[attribute=value]**
- **[attribute!=value]**
- **[attribute^=value]**
- **[attribute$=value]**
- **[attribute*=value]**
- **[attribute|=value]**
- **[attribute~=value]**
- **[attribute]**
- **[attribute2]**

Forms

- **:input**
- **:text**
- **:password**
- **:radio**
- **:checkbox**
- **:submit**
- **:image**
- **:reset**
- **:button**
- **:file**

Form Filters

- **:enabled**
- **:disabled**
- **:checked**
- **:selected**

Core

jQuery function

```
$
    .jQuery( selector [, context] ) | element | elementArray | jQueryObject
    .jQuery( )
$ .jQuery( html [, owner] ) | html, props )
$ .jQuery( fn )
def .when(deferreds)
fn jQuery.sub( )
$ .holdReady( hold )
```

jQuery Object Accessors

```
$ .each( fn(index, element) )
num .size( ), length
str .selector
el .context
$ .eq( index )
    jQuery.error( str )
[el].el .get( [index] )
num
    .index( ), .index( selector | element )
    $.jQuery.pushStack( elements, [name, args] )
arr .toArray( )
```

Interoperability

```
$.jQuery.noConflict( [extreme] )
```

http://futurecolors.ru/jquery/

Attributes

Attributes

```
str
    .attr( name | name , value )
$   .attr( name, val | map | name, fn(index, attr) )
$ .removeAttr( name )
$ .prop( name )
$ .removeProp( name )
```

Class

```
$
    .addClass( class | fn(index, class) )
bool .hasClass( class )
$
    .removeClass( [class] | fn(index, class) )
$
    .toggleClass( class [, switch] | fn(index, class) [, switch] )
```

HTML, text

```
str .html( )
$
    .html( val | fn(index, html) )
str .text( )
$
    .text( val | fn(index, html) )
```

Value

```
str/arr .val( )
$
    .val( val | fn() )
```

AJAX

Low-Level Interface

```
jqXHR jQuery.ajax( options, [settings] )
    jQueryObject accepts                bool async = true
        fn beforeSend( jqXHR, config)   bool cache = true
        fn complete( jqXHR, status)     map contents
        str contentType                 obj context
        map converters                  bool crossDomain
        obj, str data                   fn dataFilter( data, type )
        bool global = true              map headers
        bool ifModified = false         str jsonp
            fn jsonpCallback            str password
        bool processData = true         str scriptCharset
        map statusCode                  num timeout
        bool traditional                str type = 'GET'
            str url = curr. page        str username
            fn xhr
            str dataType = ( xml, json, script, html)
            fn error( jqXHR, status, errorThrown )
            fn success( data, status, jqXHR )
    jQuery.ajaxSetup( options )
```

Miscellaneous

```
str .serialize( )
[obj] .serializeArray( )
str jQuery.param( obj, [traditional] )
```

Shorthand Methods

```
$
    .load( url [, data] [, fn( responseText, [textStatus, jqXHR] )] )
jqXHR
    jQuery.get( url [, data] [, fn( data, status, [jqXHR] )] [, type] )
jqXHR
    jQuery.getJSON( url [, data] [, fn( data, status, [jqXHR] )] )
jqXHR jQuery.getScript( url [, fn( data, status )] )
jqXHR
    jQuery.post( url [, data] [, fn( data, status )] [, type] )
```

Global Ajax Event Handlers

```
$
    .ajaxComplete( fn( event, XHR, options ) )
$
    .ajaxError( fn( event, XHR, options, thrownError ) )
$ .ajaxSend( fn( event, XHR, options ) )
$ .ajaxStart( fn( ) )
$ .ajaxStop( fn( ) )
$ .ajaxSuccess( fn(event, XHR, options) )
```

CSS

CSS

```
str .css( name )
$
    .css( name, val | map | name, fn(index, val) )
```

Positioning

```
obj .offset( )
$ .offset( coord | fn( index, coord ) )
$ .offsetParent( )
obj .position( )
int .scrollTop( )
$ .scrollTop( val )
int .scrollLeft( )
$ .scrollLeft( val )
```

Height and Width

```
int .height( )
$ .height( val | fn(index, height ) )
int .width( )
$
    .width( val | fn(index, height ) )
int .innerHeight( )
int .innerWidth( )
int .outerHeight( [includeMargin] )
$ .outerHeight( val | fn(index, outerHeight ) ) 1.8+
int .outerWidth( [includeMargin] )
$ .outerWidth( val | fn(index, outerWidth ) ) 1.8+
```

Events

Page Load

```
$ .ready( fn() )
```

Event Handling

```
$ .on( events [, selector] [, data], handler ) 1.7+
$ .on( events-map [, selector] [, data] ) 1.7+
$ .off( events [, selector] [, handler] ) 1.7+
$ .off( events-map [, selector] ) 1.7+
$ .bind( type [, data ], fn(eventObj) )
$ .bind( type [, data], false )
$ .bind( array )
$ .unbind( [type] [, fn])
$ .one( type [, data ], fn(eventObj) )
$ .trigger( event [, data])
obj .triggerHandler( event [, data])
$ .delegate( selector, type, [data], handler)
$
    .undelegate( [selector, type, [handler]] ) | selector, events | namespace
```

Live Events

```
$ .live( eventType [, data], fn() )
$ .die( ), .die( [eventType] [, fn] )
```

Interaction Helpers

```
$ .hover( fn(eventObj), fnOut(eventObj) )
$ .toggle( fn(eventObj), fn2(eventObj) [, ...])
```

Event Helpers

```
function ( [data,] [fn] )
$ .blur,
  .change,
  .click,
  .dblclick,
  .error,
  .focus,
  .focusin,
  .focusout,
  .keydown,
  .keypress,
  .keyup,
  .load( [data,] fn ),
  .mousedown,
  .mouseenter,
  .mouseleave,
  .mousemove,
  .mouseout,
  .mouseover,
  .mouseup,
  .resize,
  .scroll,
  .select,
  .submit,
  .unload( [data,] fn )
```

Traversing

Filtering

```
$ .eq( index )
$ .first( )
$ .last( )
$ .has( selector ), .has( element )
$ .filter( selector ), .filter( fn(index) )
bool
    .is( selector | function(index) | jQuery object | element ) 1.3+
$ .map( fn(index, element) )
$ .not( selector ), .not( elements ),
    .not( fn( index ) )
$ .slice( start [, end] )
```

Tree traversal

```
$ .children( [selector] )
$
    .closest( selector [, context] | jQuery object | element )
arr .closest( selectors [, context] ) removed
$ .find( selector | jQuery object | element )
$ .next( [selector] )
$ .nextAll( [selector] )
$ .nextUntil( [selector] )
$ .offsetParent( )
$ .parent( [selector] )
$ .parents( [selector] )
$ .parentsUntil( [selector] )
$ .prev( [selector] )
$ .prevAll( [selector] )
$ .prevUntil( [selector] )
$ .siblings( [selector] )
```

Miscellaneous

```
$ .add( selector [, context] | elements | html )
$ .andSelf( )
$ .contents( )
$ .end( )
```

Manipulation

Inserting Inside

```
$ .append( content | fn( index, html ) )
$ .appendTo( target )
$ .prepend( content | fn( index, html ) )
$ .prependTo( target )
```

Inserting Outside

```
$ .after( content | fn() )
$ .before( content | fn() )
$ .insertAfter( target )
$ .insertBefore( target )
```

Inserting Around

```
$ .unwrap( )
$ .wrap( wrappingElement | fn )
$ .wrapAll( wrappingElement | fn )
$ .wrapInner( wrappingElement | fn )
```

Replacing

```
$ .replaceWith( content | fn )
$ .replaceAll( selector )
```

Removing

```
$ .detach( [selector] )
$ .empty( )
$ .remove( [selector] )
```

Copying

```
$
    .clone( [withDataAndEvents], [deepWithDataAndEvents] )
```

Effects

Basics

```
$ .show( [ duration [, easing] [, fn] ] )
$ .hide( [ duration [, easing] [, fn] ] )
$ .toggle( [showOrHide] )
$ .toggle( duration [, easing] [, fn] )
```

Sliding

```
$ .slideDown( duration [, easing] [, fn] )
$ .slideUp( duration [, easing] [, fn] )
$ .slideToggle( [duration] [, easing] [, fn] )
```

Fading

```
$ .fadeIn( duration [, easing] [, fn] )
$ .fadeOut( duration [, easing] [, fn] )
$ .fadeTo( [duration,] opacity [, easing] [, fn] )
$ .fadeToggle( [duration,] [, easing] [, fn] )
```

Custom

```
$ .animate( params [, duration] [, easing] [, fn] )
$ .animate( params, options )
$ .stop( [queue] [, clearQueue] [, jumpToEnd] ) 1.7+
$ .delay( duration [, queueName] )
```

Settings

```
bool jQuery.fx.off
num jQuery.fx.interval
```

Event object

```
event = {
    el currentTarget,
     * data,
    bool isDefaultPrevented( ),
    bool
        isImmediatePropagationStopped( ),
    bool isPropagationStopped( ),
     str namespace,
    num pageX,
    num pageY,
        preventDefault( ),
     el relatedTarget,
    obj result,
        stopImmediatePropagation( ),
        stopPropagation( ),
     el target,
    num timeStamp,
     str type,
     str which
}
```

Deferred

```
deferred object = {
    def
        .always( alwaysCallbacks [, alwaysCallbacks] )
    def done( doneCallbacks )
    def fail( failCallbacks )
    bool isRejected( ) deprecated
    bool isResolved( ) deprecated
     str state( ) 1.8+
    def .notify( args ) 1.7+
    def
        .notifyWith( context, [args] ) 1.7+
    def
        .pipe( [doneFilter] [, failFilter] [, progressFilter] ) 1.7
    def
        .progress( progressCallbacks ) 1.7+
    def reject( [args] )
    def rejectWith( context, [args] )
    def resolve( [args] )
    def resolveWith( context, [args] )
     str state( )
    def
        then( doneCallbacks, failCallbacks [, progressCallbacks] ) 1.7+
}
def .promise( [target] )
```

Callbacks

```
callbacks object = {  1.7+
    und .add( callbacks )
    und .disable( )
    und .empty( )
    und .fire( arguments )
    bool .fired( )
    und
        .fireWith( [context] [, args] )
    bool .has( callback )
    und .lock( )
    bool .locked( )
    und
        .remove( callbacks )
}
cb $.Callbacks( flags )
```

Utilities

Browser and Feature Detection

```
obj jQuery.support
obj jQuery.browser deprecated
str jQuery.browser.version deprecated
bool jQuery.boxModel deprecated
```

Basic operations

```
obj jQuery.each( obj, fn( i, valueOfElement ) )
obj
    jQuery.extend( [deep,] target, obj1 [, objN] )
arr jQuery.grep( arr, fn( el, i ) [, invert] )
arr jQuery.makeArray( obj )
arr jQuery.map( arrayOrObject, fn( el, i ) )
num jQuery.inArray( val, arr )
arr jQuery.merge( first, second )
 fn jQuery.noop
 fn
    jQuery.proxy( fn, scope | scope, name )
arr jQuery.unique( arr )
str jQuery.trim( str )
obj jQuery.parseJSON( str )
```

Data functions

```
$ .clearQueue( [name] )
$ .dequeue( [name] ),
  jQuery.dequeue( [name] )
obj jQuery.data( el, key ),
  jQuery.data( )
obj .data( ), .data( key )
$ .data( key, val | obj )
$
    .removeData( [name] | [list] ) 1.7+
[fn] .queue( [name] )
  jQuery.queue( [name] )
$ .queue( [name,] fn( next ) ),
  jQuery.queue( [name,] fn( ) )
$ .queue( [name,] queue ),
  jQuery.queue( [name,] queue )
```

Test operations

```
 str jQuery.type( obj )
bool jQuery.isArray( obj )
bool jQuery.isEmptyObject( obj )
bool jQuery.isFunction( obj )
bool jQuery.isPlainObject( obj )
bool jQuery.isWindow( obj )
bool jQuery.isNumeric( val ) 1.7+
```

Rappels sur Canvas

Étant donné un élément canvas conservé dans la variable cvs, on en obtient le contexte graphique avec `ctx=cvs.getContext("2d")`.

Spécification de la couleur d'un dessin: `ctx.strokeStyle="#...";ctx.strokeWidth=...`

Dessin d'une ligne entre `x1,y1` et `x2,y2`:

```
ctx.beginPath();
ctx.moveTo(x1,y1);
ctx.lineTo(x2,y2);
ctx.stroke();
ctx.endPath();
```

Une implantation de la fonction javascript `fini`

```
var nb=0;
function fini(jeu){
    for(var k=0;k<3;k++){
        // horizontal
        if(jeu[k][0]!="" && jeu[k][0]==jeu[k][1] && jeu[k][0]==jeu[k][2]){
            return {message:jeu[k][0] + " a gagné",ligne:[k,0,k,2]};
        }
        // vertical
        if(jeu[0][k]!="" && jeu[0][k]==jeu[1][k] && jeu[0][k]==jeu[2][k]){
            return {message:jeu[0][k] + " a gagné",ligne:[0,k,2,k]};
        }
    }
    // diagonal
    if(jeu[0][0]!="" && jeu[0][0]==jeu[1][1] && jeu[0][0]==jeu[2][2]){
        return {message:jeu[0][0] + " a gagné",ligne:[0,0,2,2]};
    }
    if(jeu[0][2]!="" && jeu[0][2]==jeu[1][1] && jeu[0][2]==jeu[2][0]){
        return {message:jeu[0][2] + " a gagné",ligne:[0,2,2,0]};
    }
    nb++;
    if(nb==9) return {message:"partie nulle",ligne:[]};
    return null;
}
```