Learning Javascript using the game PEXESO – PAIRS

PEXESO – PAIRS

This page describes the game Pairs.

The game **Pairs** is written in **JavaScript** and standard **HTML** commands.

The game is based on the principle of popular desk game – Pairs. The player´s task is to find the two same cards (or two cards with the same topic in the version Pairs Mix).
The player can click on the card and then he can see the picture. When he clicks on the two cards with the same symbol, the cards will stay displayed. When the two cards are different, there will be turned back.

The game begins by click on **Start** and can be finished by click on **Stop.**

Before the game starts, the player can set up the **level 0-9;** the level impacts the time of cards display. During the game the player can see the measured time, number of attempts and number of founded pairs.

Both versions of the game display the IT topics – free and paid software. In the version of „Pairs“ the player is looking for the same symbols, in the version of „Pairs Mix“ the player must find the same function of the software in free and paid version (example of the mixed pair is the text editor:

Microsoft Word (paid software) – Open Office Writer (open source)).

The "**Teachers Guide**" button will show a brief description of the game and its purpose within the Ingot project.

Game Download

The game can be copied and modified according to your own skills:

right clicking the mouse will bring up a menu. The "**View Source**" item on the menu must be selected (the text of the menu can differ slightly according to the browser used; the text "**View Source**" refers to the Internet Explorer browser and the "**View Page Source"** is used by the Mozilla Firefox browser).

Mark the text using standard **Windows OS** commands (e.g. **CTRL+A**) and copy it into the clipboard (e.g. **CTRL+C**). Open any text editor capable of processing plain text (e.g. **Notepad**, which is part of Windows OS) and paste the text into the editor from the clipboard (e.g. **CTRL+V**).

Save the text displayed in the word editor as a file of any name and use either "**html**" or "**htm**" extension, which is the standard for the files to be opened in any internet browser.

Javascript code

Javascript code is saved separately in the file **pexeso.js.**

There is possible to download the file by this way, that in the address line of the browser there is deleted **„pexeso.htm“** (eventually pexeso1.htm) and replaced by **„pexeso.js“**

Images are another essential part of the game. There are saved in the sub-folder images.

The list is as follows:

Common pictures:

0.gif, 1.gif , 2.gif , 3.gif , 4.gif , 5.gif, 6.gif, 7.gif, , 8.gif, 9.gif, 10.gif – images of figures and backgrounds

plusminus1.gif, plusminus2.gif, plusminus3.gif, plusminus4.gif – pictures of symbols **plus** and **minus**

startstop1.gif, startstop2.gif, startstop3.gif, startstop4.gif – pictures of buttons**start** and **stop**

attempts.gif, found.gif, level.gif, s.gif, time.gif – pictures of titles

blank.gif, – background picture

card.png, – picture of reverse of the card

Cards for the version **Pairs**

pic1.jpg , pic2.jpg , pic3.jpg, pic4.jpg , pic5.jpg , pic6.jpg, pic7.jpg , pic8.jpg – cards pictures for the version Pairs

Cards for the version **Pairs Mix**

pic1a.jpg , pic2a.jpg , pic3a.jpg, pic4a.jpg , pic5a.jpg , pic6a.jpg, pic7a.jpg , pic8a.jpg
pic1b.jpg , pic2b.jpg , pic3b.jpg, pic4b.jpg , pic5b.jpg , pic6b.jpg, pic7b.jpg , pic8b.jpg – cards pictures for the version Pairs Mix

The images can be downloaded via changing the address in the browser bar:
„pexeso.htm“ (eventually pexeso1.htm) is replaced with for example **„images/0.gif“**

(example: www.ingot.org/javascript/domino/pexeso**.htm** [1]to
www.ingot.org/javascript/pexeso/images/0.g [2]if)

After pressing "**Enter**" on the keyboard, a dialog box will prompt you to save a file. The file must be saved under the same name and extension and into the same directory containing the previous file.

The image must be saved in the subdirectory named "**images**", which must be created beforehand.

Correct function of all downloaded components of the game can be verified by launching the**pexeso.htm** (or **pexeso1.htm**) file.

Replacing Images

The game images can be replaced with your own ones. The game background can be modified under the following conditions:

- the image files must have an identical **name** and **extension**(more experienced programmers can use a different type of the image file, provided they modify its source code accordingly as described below)
- the game background is only a template, which consists of its repeated parts
- the game background must be transparent (description of this feature is outside of the scope of this manual and can be found on the web)
- the images must have an identical length and width measured in pixels (description of this feature is outside of the scope of this manual and can be found on the web)

Changes of the Directory Name, Using Different Image File Formats

If the different names must be used for image files or a subdirectory containing them, it is necessary to modify them properly by editing the **pexeso.htm** file. This action is designed for more experienced programmers.

In the file of **pexeso.js** the line: var IMG_PATH = "images/"; must be found.

The expression in the bar can be replaced by another directory where will be the images stored. The directory must be a sub-directory of a folder where are the files **pexeso.htm** (eventually pexeso1.htm) and **pexeso.js**

All expressions "**gif**" present in the code must be replaced with your own ones.The expression "**gif**" is the file extension of the image files used. The image file format can be replaced by extensions "**png**" or "**jpg**" (except for the background, which must be transparent – description of this feature is outside of the scope of this manual and can be found on the web).

Brief Description of the Game Functions in JavaScript Language

-->

As mentioned earlier, the game is written using **JavaScript** and **HTML** commands. Brief help descriptions for functions and elements, and descriptions of variables are stated directly in the code (in the comment lines). In order to understand the function, basic knowledge of **JavaScript** and **HTML** programming is necessary.

Html code for display of graphical text items, links and buttons is present in the file **pexeso.htm** (eventually pexeso1.htm in Pairs Mix)

Javascript code:

Definition of variables

```
// Game settings, check out pexeso.htm (line 8)
 // <body bgcolor="#FFFF99" onload="loadImages(8, 0, 0);">
 //                                 |  |  |
 //                                 |  |  |-- highscore mode (js/php)
 //                                 |  |----- game mode, see above
 //                                 |-------- count of image pairs
 //
```

Setting up the game version:

For the version Pairs the mode 0 is set.

(Setting is provided in the file pexeso.htm, line: <BODY onload="loadImages(8, 0, 0);" >)

For the version Pairs Mix the mode 1 is set.

(Setting is provided in the file pexeso1.htm line: <BODY onload="loadImages(8, 1, 0);" >)

```
 // game mode 0 : match of two identical pictures,
 //             filenames pic1.jpg, pic2.jpg ...
 // game mode 1 : match of two similar pictures,
 //             filenames pic1a.jpg, pic1b.jpg, pic2a.jpg, pic2b.jpg ...
 //
```

Here the directory for images can be set:

```
 var IMG_PATH   = "images/";                // Path of the pictures
 var arrHighScore    = new Array();          // Array of highscore objects
 var imgBackside     = new Image();          // The backside of a card
 var imgArrStartStop = new Array(4);         // Start and stop button
 var imgArrPlusMinus = new Array(4);         // Plus and minus button
 var imgArrNumber    = new Array(11);        // Ciphers 0 to 9
 var bRunning        = false;                // State of the game
 var nLevel          =  4;                   // Speed level
 var nSeconds        =  0;                   // Duration of the game in seconds
```

```
var nAttempts       =  0;                  // Number of attempts
var nHit           =  0;                 // Number of hits
var nSelected1      = -1;                  // First selected picture
var nSelected2      = -1;                  // Second selected picture
var bShowCard       = false;               // State of the card (uncovered or not)
var strPlayerName   = "xxx";               // Name of player for highscore
var strDate         = "";                 // Date for highscore
var nPoints         = 0;                  // Points for highscore
var bCookies        = false;               // Are cookies enabled
var nHighscoreMode  = 0;                   // 0=Cookies 1=PHP

var IMG_START, IMG_STOP, IMG_PLUS, IMG_MINUS;
var IMG_LEVEL, IMG_SEC, IMG_ATTEMPTS, IMG_HIT
var IMG_OFFSET, IMG_MODE, imgArrField, nImages, nSumImages;
```

Function for uploading the images:

```
/*
 *  Preload of images
 */

function loadImages(nPairs, nImageMode, nScoreMode)
{
  nHighscoreMode = nScoreMode;
  IMG_MODE = nImageMode;
  nImages = nPairs;
  imgArrField = new Array(nImages * 2);
  nSumImages  = nImages * 2 + 4 + 4 + 11;

  IMG_START = 2 + nImages * 2;
  IMG_STOP  = 4 + nImages * 2;
  IMG_PLUS  = 12 + nImages * 2;
  IMG_MINUS = 8 + nImages * 2;
  IMG_LEVEL = 10 + nImages * 2;
  IMG_SEC   = 17 + nImages * 2;
  IMG_ATTEMPTS = 25 + nImages * 2;
  IMG_HIT   = 33 + nImages * 2;

  if(document.images)
  {
    imgBackside.src  = IMG_PATH + "card.png";

    // Load start and stop button
    for(var i = 0; i < 4; i++)
    {
      imgArrStartStop[i] = new Image();
      imgArrStartStop[i].src = IMG_PATH + "startstop" + (i + 1) + ".gif";
    }

    // Load plus and minus button
    for(var i = 0; i < 4; i++)
    {
      imgArrPlusMinus[i] = new Image();
      imgArrPlusMinus[i].src = IMG_PATH + "plusminus" + (i + 1) + ".gif";
    }

    // Load ciphers
```

```
    for(var i = 0; i < 11; i++)
    {
      imgArrNumber[i] = new Image();
      imgArrNumber[i].src = IMG_PATH + i + ".gif";
    }

    // Load  pictures
    for(var i = 0; i < nImages; i++)
    {
      if(IMG_MODE == 1)
      {
        imgArrField[i * 2] = new Image();
        imgArrField[i * 2].src = IMG_PATH + "pic" + (i + 1) + "a.jpg";
        imgArrField[i * 2 + 1] = new Image();
        imgArrField[i * 2 + 1].src = IMG_PATH + "pic" + (i + 1) + "b.jpg";
      }
      else
      {
        IMG_MODE = 0;
        imgArrField[i * 2] = new Image();
        imgArrField[i * 2].src = IMG_PATH + "pic" + (i + 1) + ".jpg";
        imgArrField[i * 2 + 1] = new Image();
        imgArrField[i * 2 + 1].src = IMG_PATH + "pic" + (i + 1) + ".jpg";
      }
    }



    nLevel = 4;
    nSeconds  =  0;
    nAttempts =  0;
    nHit      =  0;
    searchFirstImage();
    clearField();
    updateAll();
    setTimeout("checkLoading()", 1000)
  }
}
```

Function for shuffling the cards:

```
/*
*  Shuffle all  pictures
*/

function shuffle()
{
  if(document.images)
  {

    // Swap two pictures (random index)
    var j = Math.floor(new Date().getSeconds() * Math.random() + 60);
    for(var i = 0; i < j; i++)
    {
      n1 = Math.round(Math.random() * (nImages * 2 - 1));
      n2 = Math.round(Math.random() * (nImages * 2 - 1));
      img = imgArrField[n1];
```

```
      imgArrField[n1] = imgArrField[n2];
      imgArrField[n2] = img;
    }
  }
 }
```

Start game:

```
/*
 *  Start new game
 */

 function startGame()
 {
  if(document.images)
  {
   if(!bRunning)
   {
    shuffle();
    clearField();
    nSeconds = 0;
    nSelected1 = -1;
    nSelected2 = -1;
    nAttempts  =  0;
    nHit       =  0;
    id = setInterval("countSeconds()", 1000)
    bRunning = true;
    bShowCard = false;
    updateAll();
   }
  }
 }
```

Stop game:

```
 /*
 *  Stop game
 */

 function stopGame()
 {
  if(document.images)
  {
   if(bRunning)
   {
    clearInterval(id);
    bRunning = false;
    updateAll();
   }
  }
  return;
 }
```

Time measuring:

```
/*
 *  Count seconds
 */
```

```
function countSeconds()
{
  nSeconds++;
  showNumber(nSeconds, IMG_SEC + IMG_OFFSET, 5);
}
```

Function for figures display:

```
/*
 *  Show numbers
 */

function showNumber(nNumber, nPosition, nCount)
{
  if(document.images)
  {
    nNumber += "";
    while(nNumber.length < nCount) nNumber = " " + nNumber;
    for(var i = 0; i < nCount; i++)
    {
      var n = nNumber.charAt(i);
      if(n == " ")
      {
        document.images[nPosition + i].src = imgArrNumber[10].src;
      }
      else
      {
        document.images[nPosition + i].src = imgArrNumber[n].src;
      }
    }
  }
}
```

Function for the reverse side of cards display:

```
/*
 *  Set the backside of all cards
 */

function clearField()

{
  if(document.images)
  {
    for(var i = 0; i < nImages * 2; i++)
    {
      document.images[i + IMG_OFFSET].src = imgBackside.src;
    }
  }
}
```

Function for display the selected cards:

-->

```
/*
 *  Show card
 */

 function showCard(nImage)

 {
  if(document.images)
  {
   if(bRunning && !bShowCard)
   {

    // Uncover a card, if not allready two are selected
    if(nSelected1 == -1 || nSelected2 == -1)
    {
     // Is the clicked not uncovered ?
     if(document.images[nImage + IMG_OFFSET].src == imgBackside.src)
     {

      // Uncover selected card
      document.images[nImage + IMG_OFFSET].src = imgArrField[nImage].src;
      if(nSelected1 == -1)
      {
       nSelected1 = nImage;
      }
      else
      {
       nSelected2 = nImage;
      }
     }
    }



    // There are two cards uncovered
    if(nSelected1 != -1 && nSelected2 != -1)
    {
     showNumber(++nAttempts, IMG_ATTEMPTS + IMG_OFFSET, 5);



     // Are the cards equal ?
     var pic1 = document.images[nSelected1 + IMG_OFFSET].src;
     var len1 = pic1.length;
     var pic2 = document.images[nSelected2 + IMG_OFFSET].src;
     var len2 = pic2.length;
     if(pic1.substr(0, len1 - IMG_MODE - 4) == pic2.substr(0, len2 - IMG_MODE - 4))

     {
      // If there are equal, incement the counter
      showNumber(++nHit, IMG_HIT + IMG_OFFSET, 5);
      nSelected1 = -1;
      nSelected2 = -1;



      // Are all pictures uncovered ?
      if(nHit == nImages)
      {
```

```
          stopGame();
          nPoints = Math.round(100000 * (nLevel + 1) / nSeconds / nAttempts);
          strMsg = "Tvoje skˇre je " + nPoints + " bod¨.";
          window.alert(strMsg);
         }
        }
        else
        {

         // The two cards are not identical, start now the timer for hide the cards
         bShowCard = true;
         setTimeout("clearCard()", 2000 - nLevel * 200);
        }
       }
     }
     else
     {
      if(!bRunning)
      {
       alert("Press key Start !");
      }
     }
   }
 }
```

Function for turns back the cards which are not the same:

```
/*
 *  Turn the uncovered cards
 */

 function clearCard()

 {
  document.images[nSelected1 + IMG_OFFSET].src = imgBackside.src;
  document.images[nSelected2 + IMG_OFFSET].src = imgBackside.src;
  nSelected1 = -1;
  nSelected2 = -1;
  bShowCard = false;
 }
```

Function for setting up the level:

```
/*
 *  Set the level
 */

 function setLevel(nValue)

 {
  if(document.images && !bRunning)
  {
   nLevel += nValue;
   if(nLevel < 0) nLevel = 0;
   if(nLevel > 9) nLevel = 9;
   showNumber(nLevel, IMG_LEVEL + IMG_OFFSET, 1);
```

```
  }
 }
```

Update all counters:

```
/*
 *  Update all counter
 */

 function updateAll()

 {
  if(document.images)
  {

    showNumber(nLevel, IMG_LEVEL + IMG_OFFSET, 1);
    showNumber(nSeconds, IMG_SEC + IMG_OFFSET, 5);
    showNumber(nAttempts, IMG_ATTEMPTS + IMG_OFFSET, 5);
    showNumber(nHit, IMG_HIT + IMG_OFFSET, 5);
    if(bRunning)
    {
      document.images[IMG_START + IMG_OFFSET].src = imgArrStartStop[1].src;
      document.images[IMG_STOP + IMG_OFFSET].src  = imgArrStartStop[2].src;
      document.images[IMG_PLUS + IMG_OFFSET].src  = imgArrPlusMinus[1].src;
      document.images[IMG_MINUS + IMG_OFFSET].src = imgArrPlusMinus[3].src;
    }
    else
    {

      document.images[IMG_START + IMG_OFFSET].src = imgArrStartStop[0].src;
      document.images[IMG_STOP + IMG_OFFSET].src  = imgArrStartStop[3].src;
      document.images[IMG_PLUS + IMG_OFFSET].src  = imgArrPlusMinus[0].src;
      document.images[IMG_MINUS + IMG_OFFSET].src = imgArrPlusMinus[2].src;
    }
  }
 }
```

Removal all white spaces in the word:

```
/*
 *  Remove all white spaces
 */

 function strTrim(str)
 {
  var strReturn = "";



  for(var i = 0; i < str.length; i++)
  {
    if(str.charAt(i) != " ")
    {
      strReturn += str.charAt(i);
```

```
    }
   }
   return strReturn;
 }
```

Setting up the initial zero:

```
/*
 *  Add leading zero
 */

 function addLeadingZero(value, nTotalLength)
 {
   value += "";
   while(value.length < nTotalLength) value = "0" + value;
   return value;
 }
```

Search for the first cards index:

```
/*
 *  Search for index of first image
 */

 function searchFirstImage()
 {
   for(var i = 0; i < document.images.length; i++)
   {
     if(document.images[i].name == "memory_id")
     {
       IMG_OFFSET = i + 1;
       break;
     }
   }
 }
```

The counter of played cards:

```
/*
 *  Count of preloaded images
 */

 function countLoadedImages()
 {
   var nCompleted = 0;
   for(var i = 0; i < 2 * nImages; i++)
   {
     if(imgArrField[i].complete)
     {
       nCompleted++;
     } //else alert(imgArrField[i].src);
  }

    for(var i = 0; i < 4; i++)
   {
     if(imgArrStartStop[i].complete)
```

```
    {
      nCompleted++;
    } //else alert(imgArrStartStop[i].src);
  }

  for(var i = 0; i < 4; i++)
  {
    if(imgArrPlusMinus[i].complete)
    {
      nCompleted++;
    } //else alert(imgArrPlusMinus[i].src);
  }

  for(var i = 0; i < 11; i++)
  {
    if(imgArrNumber[i].complete)
    {
       nCompleted++;
    } //else alert(imgArrNumber[i].src);
  }
  return nCompleted;
 }
```

Indication of images uploads:

```
/*
 *  Progressbar (state of preloading images)
 */

function checkLoading()
{
  var nLoaded = countLoadedImages();
  if(nLoaded < nSumImages)
  {
    document.images[IMG_OFFSET - 1].width = 456 - (456 / nSumImages * nLoaded);
    window.status = "" + nLoaded + " of " + nSumImages + " pictures loaded";
    setTimeout("checkLoading()", 250);
  }
  else
  {
    document.images[IMG_OFFSET - 1].src = IMG_PATH + "blank.gif";
    document.images[IMG_OFFSET - 1].width = 5;
    window.status = "";
    updateAll();
  }
}
```

Source URL: https://theingots.org/community/pairs

Links
[1] http://www.ingot.org/javascript/domino/pexeso.htm
[2] http://www.ingot.org/javascript/pexeso/images/0.g