



2012 Provincial Skills Canada Competition

Scope Document

Edmonton Expo Centre, Edmonton

May 10 & 11, 2012

EVENT: Web Site Development	LEVEL: Secondary
START TIME May 10 th : 8:00 am – 12pm, 1pm-4:30pm May 11 th : 7:30 am – 12pm	LOCATION: Hall B, Edmonton Expo Centre, Edmonton
DURATION: 12 hrs. (Two Days)	REGIONALIZED EVENT: No
WORLDSKILLS TRADE #: 17	

GENERAL DESCRIPTION

Purpose of the Challenge:

To provide an opportunity for students to demonstrate their ability to design a dynamic and aesthetically pleasing website.

Skills and Knowledge:

- Ability to develop web pages / a web site on a fictional scenario that is outlined below.
- Some features that competitors may include on their web pages are: menus, graphics, forms, email, tables, styles, internal and external links, and bullets.
- The majority of the site theme needs to be built using XHTML. Presentation formatting should be done with CSS.
- A proper planning process must be presented on paper (storyboard / flowchart).
- Websites should be functional and visually pleasing in various screen resolutions.
- Site should be easy to maintain and user friendly.
- Cross browser compatibility between latest versions of Firefox, Safari, and Internet Explorer should be tested.
- Software will be given to test your site locally.
- All final project media must be uploaded to the server for testing (Instructions will be given). Software will be given to test locally as well.

Optional elements worth 25% of total mark:

- Use of a database
- Server side script
- Client side script
- Animation

Look to the appendices at the end of this document for various examples of these items.

SKILLS20

EQUIPMENT & MATERIALS

Equipment and Materials Competitors Must Supply:

- Appropriate materials for planning (i.e.: paper, pen, & ruler).

Equipment and Materials Supplied by the Committee:

Hardware:

- Windows based computer with monitor and mouse. Specs not available at this time.

Software:

- Base System
 - Windows 7
 - MS Office
- Text Editors / Design Tools (competitors have the choice of these tools)
 - Dreamweaver
 - FrontPage (FrontPage Extensions not supported)
 - HTML Kit
 - Notepad++
 - Editplus
 - UltraEdit
 - Textpad
 - Metapad
 - PHP Designer
 - Vim Colour Editor
- Graphics
 - Adobe Creative Suite
 - Flash
 - Freehand
 - Fireworks
 - Paintshop Pro
 - GIMP
- Testing
 - Firefox and Internet Explorer
 - XHTML Validation Tool
 - CSS Validation Tool
- Server Based Applications
 - XAMPP (installed locally)
 - Apache
 - PHP
 - MySQL database
 - phpMyAdmin

SKILLS20

- PuTTY
- PHP / MySQL Reference from their respective sites will be available on the workstations

- * Registered Global Variables will be off for the PHP server.
- * Provincial Competition's software is in line with the National Competition's server and client side software.
- * We do not guarantee software versions.

Clothing Requirement

Appropriate business attire should be worn.

SAFETY

The health, safety and welfare of all individuals involved with Skills Canada Alberta are of vital importance. Safety is a condition of participation with Skills Canada Alberta and shall not be sacrificed for the sake of expediency. At the discretion of the judges and technical committees, any competitor can be denied the right to participate should they not have the required proper safety equipment and/or act in an unsafe manner that can cause harm to themselves or others.

JUDGING

Criteria: (out of 1000 pts.)

- Storyboard/Visual/Design / 350
- Technical elements / 350
- Optional Elements (Choose one or more) /200
 - Database
 - Animation
 - Script (Server side or Client side)
- Presentation of website to judges (required to answer basic client type questions) / 100

RELATED CAREER AND TECHNOLOGY STUDIES MODULES

COM1005: Visual Composition

COM1055: Web Design 1

COM2035: Raster Graphics 1

COM2045: Vector Graphics 1

COM2055: Web Design 2

COM3035: Raster Graphics 2

COM3045: Vector Graphics 2

COM3055: Rich Media – Basics

COM3065: Rich Media – Programming

COM3075: Cascading Style Sheets

CSE1110: Structured Programming 1

CSE1120: Structured Programming 2

CSE1210: Client-side Scripting 1

CSE1220: Client-side Scripting 2

CSE2110: Procedural Programming 1

CSE2120: Data Structures 1

CSE2210: Client-side Scripting 3

CSE3210: Server-side Scripting 1

INF1050: Database 1

Descriptions of all modules are located at the following website:

<http://www.education.gov.ab.ca/cts/infopro/>

SKILLS20

ADDITIONAL INFORMATION

- No additional materials, hardware or software will be allowed for the contest.
- Internet access for individuals will be provided in 10 minute intervals for a maximum of 2x per day.
- CDROM/DVD/USB will be off limits during the competition.
- No Cellular Telephone devices will be allowed.
- Reference Books may be brought into the competition under the following conditions:
 - Books must be brought in at the start of the competition on day 1.
 - Books will be placed on a communal table and may not be brought to an individual workstation.
 - Books will be available to all competitors.
 - Books must be left until the end of the competition just before judging.
 - No customized notes will be allowed.
 - Books must be brought to committee staff for inspection.
 - Only commercially published books will be allowed.
 - No more than one book per competitor.
 - Book ownership must be noted.

Competitor Registration

The Provincial Skills Canada Competition (PSCC) registration will open online www.skillsalberta.com on February 27th, 2012 at 8:00 am.

Lunch

Lunch for accredited competitors only will be provided by Skills Canada Alberta. Teacher accreditation can be purchased online at www.skillsalberta.com. Registration opens on February 27, 2012 at 8:00 am.

Parking & Venue Maps

<http://www.edmontonexpocentre.com/attend/parking/>

Awards Ceremony

The awards ceremony will take place on Friday May 11, 2012 at 5:30pm in Hall D of the Edmonton Expo Center. Admission is free and everyone is welcome to attend. The awards ceremony will be shown live at www.skillsalberta.com.

Team Alberta Information

Team Alberta will be selected at the Provincial Skills Canada Competition (PSCC) on May 10-11, 2012. Then two days later the gold medalists from PSCC will compete at the Skills Canada National Competition (SCNC) in Edmonton, Alberta. It is recommended that competitors review the SCNC scope documents to be familiar with the national scope and project.

http://www.skillsalberta.com/index.php?option=com_content&task=view&id=226&Itemid=242

During the PSCC Awards Ceremony on May 11 all students who earn a medal will be escorted backstage after we photograph their onstage moments of success. Gold medalists will be given their Team Alberta information package and will confirm their participation in the SCNC.



20th ANNUAL PROVINCIAL SKILLS CANADA COMPETITION

SKILLS20

Students must be present at the Awards Ceremony to claim their position on Team Alberta. If the Gold medalist is not able to attend SCNC, the next top ranking individuals will be asked to participate.

Please prepare your students in advance to accept a position on Team Alberta and how your school will support their participation.

If a student is not able to attend the Awards Ceremony a letter confirming the student's interest in Team Alberta participation can be emailed to kariz@skillscanada.com, otherwise the Team Alberta position will be offered to the silver medalist.

COMMITTEE MEMBERS

Harm Gerding
Bruce Zawalsky
Brad Frere (**Chair**)
Corey Johnson
Brent Lauinger
Braeden Petruk

Hearthstone Consulting & Development Ltd.
Hearthstone Consulting & Development Ltd.
Lost Dog Developments Inc.
Lost Dog Developments Inc.
Opreie Real Estate Software
CtrlShiftCreate



APPENDIX DATABASE

A database is an efficient way to collect, organize, and retrieve data. By sending queries to a database server such as MySQL, we can manipulate the data within the database, as well as obtain data within it according to the patterns and specifications that fit what we want to retrieve.

A typical database is set up in the following way:

Tables:

"Tables" are used to encapsulate related pieces of data. For example, if I was storing information about students in a school, I would create a table of students within the database. This table could hold information such as:

- Names
- ID numbers
- Grade levels / Homerooms

By organizing logically related data into different tables, we make it easier to understand our database, and also easier to work with the data inside of it.

Rows/Columns:

Having a student's table in our database is a good start; but how does one store data *within* this table? This is where rows and columns (like a spreadsheet in Excel or Numbers) come into play.

In each table, the programmer specifies columns which store different pieces of data (the three types of data above would be separate columns in the Students table). Then when we want to actually add a student to the Students table, we add a new row, and fill each column in that row with the appropriate data.

Querying the Database:

At this point, we have a table called "students" and within that table, three columns named "name," "id," and "grade". In order to programmatically manipulate and utilize this data, we need to send queries to the database to instruct it regarding what to change or provide for us.

Select:

The "SELECT" keyword is used to gather information *from* the database. Selecting content from the database follows the format:

```
SELECT [list of column names] FROM [table name] [optional: any other keywords];
```

For example, if we want to collect a list of names of all students in the school, we can use the following query:

```
SELECT name FROM students;
```

Perhaps we would like to retrieve a more specific list of students – all of the students in grade 11. SELECT can be combined with various other keywords such as WHERE, to form more specific queries. To select all students in grade 11, a query as follows would do the trick:

```
SELECT name FROM students WHERE grade=11;
```

The MySQL manual provides an extensive list of keywords that can be used to form different types of SELECT queries.

Insert:

The "INSERT" keyword is used to put data into the database. This keyword is almost always followed by the "INTO" keyword which specifies which database table the data should be inserted into.

To add a new student into our database, we follow the format (square brackets are to show placeholders):



```
INSERT INTO [table name] ([list of column names]) ([values]);
```

So to insert a grade 12 student named Bob with the ID number 1272958 into the students table, the following query would be used:

```
INSERT INTO students (id,name,grade) (1272958,"Bob",12);
```

Update:

When a student advances to the next grade, it would be very inefficient to have to remove all of those student's records and re-enter them with a new grade level attached to their record. For this reason, database servers provide the "UPDATE" keyword which can be used to alter the contents of rows stored in the table.

The UPDATE format is as follows:

```
UPDATE [table name] SET [expression] WHERE [conditions];
```

So to perform the grade incrementing operation mentioned above one first needs to specify which student (or students) needs to have their grade incremented. If we want to change the grade level of a single student, we need a reference that applies to them uniquely. Their ID number is perfect for this! We will use this ID number in the "conditions" place above to single them out. (We'll use the ID number of the student we just inserted: Then we need to set their grade level to one higher than its current value. So the following query will get the job done:

```
UPDATE students SET grade=grade+1 WHERE id=1272958;
```

If, instead, it's the end of the year and (assuming everyone passed) we want to change *all* of the students into the next grade level, our query needs to be more broad. Grade 12 students shouldn't be turned into grade 13 students (it doesn't make sense in Alberta!) so we want any student whose grade level is *less* than grade 12 to be incremented. MySQL's arithmetic operators are good for this:

```
UPDATE students SET grade=grade+1 WHERE grade<12;
```

There are *many* more common database keywords and functions which can be very helpful in data management for your website. A complete list, including extensive documentation, is available on the MySQL website. This is only a small example.

Tying this database logic into the logic of your program or website makes it possible to create user systems, online stores, analytic systems, etc. Learning the basics of database operation is highly recommended.

A good introduction to PHP and MySQL programming can be found here:

http://www.w3schools.com/php/php_mysql_intro.asp

APPENDIX SERVER SIDE SCRIPT

Server-side logic allows dynamic generation of pages, communication with other machines and software (including a database server) and much other functionality that facilitate the creation of a dynamic website.

The include Statement:

For a server-side script to execute, it needs to be invoked. This happens in one of two ways:

1. Either the user navigates to the php file directly:

```
http://example.com/test.php
```

Where the script contained in test.php would be executed.

2. Alternately, a script can be invoked through another script using the “include” statement. To “include” another PHP file inside of a script (thereby executing its contents) the following line can be added to your script:

```
include “test2.php”;
```

Let’s have a look at the effect of such an include statement. Imagine a test.php as follows:

```
<?php  
echo “test2.php outputs the following: ”;  
include “test2.php”;  
?>
```

Then test2.php contains:

```
<?php  
echo “These are the contents of test2.php!”;  
?>
```

The output of running test.php is:

```
test2.php outputs the following: These are the contents of test2.php!
```

This statement is useful for re-using code without writing it multiple times, for creating templates, or for separating your code into manageable files.

Transferring Data:

A critical part of making your website dynamic is the ability to accept input from users. Two standard methods of sending data to a web server are through “GET” and “POST” transfers. PHP provides access to these data through two superglobal (accessible everywhere in your program) arrays called **\$_GET** and **\$_POST**.

GET:

Data sent through the URL (or through other methods by external programs) is captured in the GET array. The following is a comparison between a URL with GET info, and the corresponding **\$_GET** array created by PHP:

```
http://example.com/test3.php?title=CoolPage
```

There are many server side scripting languages, PHP being one of them. A good place to start learning PHP Server Side scripting is here:

<http://www.w3schools.com/php/>



APPENDIX CLIENT SIDE SCRIPT

Client-Side scripting (most commonly, JavaScript) allows for a dynamic and interactive user experience on your website. Using JavaScript, the elements on the page can be adjusted, created, or removed without having to refresh or navigate to a new page. All modern browsers are also able to send data to and from servers using JavaScript.

The DOM:

JavaScript's Document Object Model (DOM) provides the interface through which we can access elements and input on web pages. Various built-in functions allow us to target, manipulate, and create/remove certain elements from the "document tree" that comprises the page.

Embedding JavaScript:

JavaScript can either live inside of the HTML page you create, or can reside in an external file which is *referenced* in HTML files where it is used.

Inline:

The *inline* method is done simply using the `<script>` tag which is usually placed between the `<head>` and `</head>` tags.:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset=UTF-8>
  <title>JavaScript Testing</title>
  <script>
    alert("Welcome!");
  </script>
</head>
<body>
  <p>Nothing to see here!</p>
</body>
</html>
```

This script would be very annoying; it simply prompts "Welcome!" to the user upon loading the page.

External Referencing:

Alternatively, this same script could be contained in an external file, [myscript.js](#), and referenced in our HTML page. The HTML page would look as follows:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset=UTF-8>
  <title>Javascript Testing</title>
  <script src=myscript.js>
  </script>
</head>
<body>
```

SKILLS20

```
<p>Nothing to see here!</p>  
</body>  
</html>
```

Notice the "src=myscript.js" that was added to the <script> tag. This directs the browser to the file that contains the script that should be executed. If, inside of myscript.js, we had written:

```
alert("Welcome!");
```

Then this alternative method would have the same effect: *Welcome!* would be alerted on the page.

A good place to start learning JavaScript is here:

http://www.w3schools.com/js/js_intro.asp