# Week 3 Tasks for Submission

This document contains a number of tasks for you to attempt.

There are three types of task:

- Tasks labelled as "no submission required, not part of portfolio"
  - They are for you to attempt and practice.
  - They provided assistance for you to develop solutions to other tasks that are part of your portfolio
- Tasks labelled as "PASS Submission Task"
  - These tasks are part of your portfolio geared towards all students
  - The solutions to these tasks must be uploaded for marking by your tutor
- Tasks labelled as "CREDIT Submission Task"
  - These tasks are part of your portfolio geared mainly towards students aiming for more than a pass grade. Of course students aiming for a pass grade may attempt and submit these tasks.
  - The solutions to these tasks must be uploaded for marking by your tutor

## Screen Capture Process

You will be asked to take screen captures of your HTML pages, JavaScript code etc.

Windows comes with an application called Snipping Tool which will perform this task.

Mac OSX has some built in keyboard shortcuts (Command-Shift-4). The image is saved on the desktop. https://www.itg.ias.edu/content/keyboard-shortcuts-capture-screen-shot-mac-os-x.

There are also products such as Jing (Win and OSX).

Try to avoid taking screen shots of the entire screen if you are only interested in part of that screen.

## Submission Process

Download the files **W03P.DOCX** and **W03C.DOCX** from Canvas.

Paste the required screen captures from the tasks below into the appropriate places within these files.

When complete, use the **File / Export** menu option to generate the files **W03P.PDF** and **W03C.PDF**

Finally log into **Doubtfire** and **submit** both files into the appropriate weekly tasks.

## File Locations:

In the following tasks you will often be asked to save a file to a **storage location**. You may use any storage location that you wish. You will not be asked to submit your .html or .js

# Template

From now on we will be using **template_upd.html** for creating Web pages so download this file and modify the footer area to contain your name and ID.

# JavaScript Tasks

### Task 1. (no submission required, not part of portfolio)

Create a HTML file named **w3a.html** based on **template_upd.html**

- Change the title to Testing output, description "getElementById and innerHTML" and update other meta tags as applicable.
- Add a script element which specified a file named w3a.js within the **src** attribute.
- Inside the article section place the following:
  ** **

Create a JavaScript file named **w3a.js** based on **template.js**

Within init():

- Create a local variable username and use prompt() statement to ask for the name of the user.
- Create a local variable output and assign document.getElementById("msg") to this variable.
- Use output.innerHTML to display a string stating "You entered the name <username>" – replace <username> with the value entered by the user.

Test w3a.html

### Task 2. (no submission required, not part of portfolio)

Within w3a.html:

- Inside the article section after the paragraph with id "msg" create 2 more empty paragraphs, one with id being "number" and another with id "square".

Within init():

- After the statement with local variable username and prompt() statement to ask for the name of the user:
  - create a local variable num and use prompt() asking the user to enter any number between 0 and 10
  - create a local variable sq and assign square of the entered number to sq
    Hint: you just need to multiply num by itself.
- After the statement displaying user name, create a local variable numberout and assign document.getElementById("number") to this variable.
- Use innerHTML property to display "The entered number is …" (replace … with the entered number from the relevant variable).
- Repeat the previous 2 steps to create a local variable squareout and assign document.getElementById("square") to this variable and then to display inside the last paragraph "The square of the number … is XXX" (replace … with the entered number from the relevant variable and XXX with its square from the variable sq).

Test w3a.html

## Task 3. (no submission required, not part of portfolio)

Create a HTML file named **w3b.html** based on **template_upd.html**

- Change the title to Testing output, description "dynamic HTML" and update other meta tags as applicable.
- Add a script element which specified a file named w3b.js within the **src** attribute.
- Inside the article section place the following:
  - Level 2 heading Calculating room size
  - Paragraph

    ** The room has width  metres and length  metres. Its size is  sq. m **

Create a JavaScript file named **w3b.js** based on **template.js**
Within init():

- Create a local variable wid and use prompt() statement to ask for the width of the room in metres.
- Create a local variable len and use prompt() statement to ask for the length of the room in metres.
- Create a local variable rmSize, calculate the room size and assign to this variable.
- Create a local variable widthOut and assign document.getElementById("width") to this variable.
- Use innerHTML property to display the width as stored in the variable wid.
- Create a local variable lengthOut and assign document.getElementById("len") to this variable.
- Use innerHTML property to display the length as stored in the variable len.
- Create a local variable sizeOut and assign document.getElementById("size") to this variable.
- Use innerHTML property to display the room size.

Test w3b.html

## Task 4. (no submission required, not part of portfolio)

Create a HTML file named **w3c.html** based on **template_upd.html**

- Change the title to Gross Pay Calculator, description "Payroll app" and update other meta tags as applicable.
- Add a script element which specified a file named w3c.js within the **src** attribute.
- Inside the article section:
  add the level 2 heading Pay slip
  Create a paragraph with id="job"
  Create a paragraph with id="payRate"
  Create a paragraph with id="hours"
  Create a paragraph with id="grossPay"

Create a JavaScript file named **w3c.js** based on **template.js**
The application should calculate gross pay based on the hourly pay rate and hours worked. The hourly rate depends on job code:

| Job Code | Rate per hour $ |
|---|---|
| X | 50 |
| Y | 100 |
| Z | 150 |
| Any other character | 0 |

Create a function named **determinePayRate**(). This function should accept 1 parameter: jobCode.
Within that function, do the following:

Create a variable **payRate**
Use the **switch** statement to assign the value to payrate based on jobCode. The function should handle case, i.e. treat X and x as the same value. Same rule applies to Y and y, Z and z.
Return **payRate**

Create a function **calcGross()** that takes 2 parameters: *rate* and *hours*. The function returns the result of multiplication of these parameters.

Within init():

Create variables jobCode, ratePerHour, hoursWorked, and gross.
Using a **prompt** statement ask the user to enter job code and store the user's response in the variable named jobCode.

Using a **prompt** statement ask the user to enter number of hours worked.
Store the user's response in the variable named hoursWorked.
Convert hoursWorked to a number using parseInt() or Number().

Call the function **determinePayRate**() and provide actual parameter value in brackets. Store the returned value in ratePerHour.

If rate per hour is not equal 0 then perform the following steps:

- Call the function **calcGross**() and provide actual parameter values in brackets. Store the returned value in gross.

- Display the following output by using document.getElementById statements and innerHTML property:
  "Job code is N" (N is the value entered by the user and stored in jobCode)
  "Rate per hour is $999.99" (999.99 is the value stored in ratePerHour)
  "Number of hours worked: 99" (99 is the value entered by the user and stored in hoursWorked)
  "Gross pay $999.99" (999.99 is the value stored in gross)

Otherwise (if rate per hour is 0) display an error message "Invalid job code" in the paragraph with id="job".

**Warning: if you are copying any data from this handout or lecture slides into your code, ensure that double quotes are straight as in the documents created using Word and PowerPoint they are often auto-replaced by smart quotes which are syntax error in JavaScript.**

Test your code by trying the following values when the web-page is loaded:
Job code X, y, Z, a, A and any valid number of hours worked.

Troubleshoot your program if necessary.

## Task 5. (no submission required, not part of portfolio)

Create a HTML file named **w3d.html** based on **template_upd.html**

- Change the title to Term Deposit Calculator, description "Term deposit app" and update other meta tags as applicable.
- Add a script element which specified a file named w3d.js within the **src** attribute.
- Inside the article section:
  add the level 2 heading Pay slip
  Create a paragraph with id="deposit"
  Create a paragraph with id="months"
  Create a paragraph with id="intRate"
  Create a paragraph with id="interest"
  Create a paragraph with id="balance"

Create a JavaScript file named **w3d.js** based on **template.js**
The application should calculate interest earned and closing balance of the term deposit based on the deposit amount and interest rate. The interest rate depends on the deposit duration:

| Term deposit duration | Interest rate |
|---|---|
| Less than 3 months | 2% |
| Between 3 and 6 months inclusive | 3% |
| More than 6 months and under 9 months | 4% |
| 9 months or more | 5% |

Create a function named **determineInterestRate**(). This function should accept 1 parameter: depositTerm.
Within that function, do the following:

- Create a variable **intRate**
- Use nested if statement to determine interest rate (complete the code below)

```
if (depositTerm < 3) {
        intRate = 0.02;
} else // means depositTerm>=3
        if (depositTerm <= 6) {
                intRate = …;
        } else // means depositTerm …
                if (…
```

- Return intRate

Create a function **calcInterest()** which accepts 3 parameters: depositAmount, interestRate and term. The function should calculate and **return interest**.
Note, the provided interest rate is annual, so you need to divide the interest by 12 to get monthly rate and then multiply by the term. For example, deposit of $1000 earning annual interest rate of 2% would earn $20 per year (1000*0.02) so if the term is 2 months we get 1000*0.02/12*2=3.3333.

Create a function **calcClosingBalance()** which accepts 2 parameters: depositAmount and interest. The function should calculate and **return balance**.

Create a function named **checkValidity**(). This function should accept 1 parameter: value and return Boolean value **true** if the value is >0 or **false** if it is not.

Within that function, do the following:

Create a variable **valid**
Write an if statement checking that value > 0 and assign valid to true if the condition is true or to false otherwise.
Return valid

Note, instead of writing this if statement, you can simply return value>0 because JavaScript will evaluate it to true or false.

Within init():

Create variables deposit, duration, intRate, interest, closingBalance.

Using a **prompt** statement ask the user to enter deposit amount and store it in the variable deposit.
Convert deposit to a number using parseInt() or Number().
Create a variable validDeposit and call the function **checkValidity()** passing deposit as a parameter. So validDeposit will be true or false depending on the value entered by the user.

Using a **prompt** statement ask the user to enter deposit duration in whole months and store it in the variable duration.
Convert duration to a number using parseInt() or Number().
Create a variable validTerm and call the function **checkValidity()** passing duration as a parameter. So validTerm will be true or false depending on the value entered by the user.

Use if to check that either validDeposit is false or validTerm is false and in this case display an error message in the paragraph with id=" deposit" using document.getElementById.
Otherwise (meaning both entries are valid) perform the following steps:

- Call the function **determineInterestRate()** passing the appropriate argument and store the returned value in intRate.
- Call the function **calcInterest()** passing the appropriate arguments and store the returned value in the variable interest.
- Call the function **calcClosingBalance()** passing the appropriate arguments and store the returned value in the variable closingBalance.
- Display the following output by using document.getElementById statements and innerHTML property (make sure all currency values are formatted to 2 decimal places:

  "Deposited amount $9999.99", where 9999.99 is replaced by the initial deposit amount

  "Deposit duration – N months", where N is the number of months entered by the user

  "Interest rate applied 9.9%" – here you need to convert interest rate to %, e.g. 0.02 should be displayed as 2.0%

  "Interest earned $999.99", where 999.99 is replaced by the interest as calculated in the application

  "Balance on maturity $9999.99", where 9999.99 is replaced by the closing balance.

Test your code by trying the following values when the web-page is loaded:

- Invalid deposit amount (e.g. letters or -10 or 0)
- Invalid term/duration (e.g. letters or 0 or -2)
- Valid values:

| Deposit | Term (months) | Interest rate | Interest | Balance |
|---|---|---|---|---|
| 1000 | 2 | 0.02 | 3.33 | 1003.33 |
| 1000 | 3 | 0.03 | 7.50 | 1007.50 |
| 1000 | 6 | 0.03 | 15.00 | 1015.00 |
| 1000 | 8 | 0.04 | 26.67 | 1026.67 |
| 1000 | 9 | 0.05 | 37.50 | 1037.50 |

## Task 6. (PASS Submission Task)

Create a **HTML** file named **w3P2.html** based on **template_upd.html**

- Change the title to "Ticket Sale Calculator", description "Pass level task" and update other meta tags as applicable including **your student name and ID.**
- Add a script element which specified a file named **w3P2.js** within the **src** attribute.
- Inside the article section:
  add the level 2 heading Tickets Order
  Create a paragraph with id="seatingArea"
  Create a paragraph with id="pricePerTicket"
  Create a paragraph with id="ticketQty"
  Create a paragraph with id="amountDue"

Create a **JavaScript** file named **w3P2.js** based on **template.js.**

We want an application that determines a ticket price based on seating area category and calculates cost based on price and quantity. No more than 20 tickets can be sold in one transaction. Transaction processing fee is flat $7.

Ticket price rule:

| Seating area category | Price per ticket $ |
|---|---|
| S (stalls) | 180 |
| C (circle) | 150 |
| B (balcony) | 100 |
| R (restricted view) | 70 |
| Any other character | 0 |

Create a function **determineTicketPrice()** that takes 1 parameter - seating area category and returns price of a ticket. Use the **switch** statement. Make sure it handles the case, i.e. S and s will result in $180, C and c will result in $150, etc.

Create a function that takes 2 parameters: price per ticket and quantity of tickets and returns total amount due (including processing fee).

Create a function **validateQty()** that takes one parameter ticketQty and returns true if this parameter is between 1 and 20 inclusive or false otherwise.

Within init():

- Create variables seatCategory, ticketPrice, ticketQty, validQty, ticketCost.
- Using a **prompt** statement ask the user to enter seating area category.
- Using a **prompt** statement ask the user to enter ticket quantity, convert it to a number, then call validateQty() and store the returned value in the created validQty variable.
- Call the function **determineTicketPrice()** with the appropriate argument and store the returned value in ticketPrice.
- Check if ticket price is 0 and if true, display an error message that seating category is invalid in the paragraph with id="seatingArea"
- Check if validQty is false and if this is the case display an error message "Invalid number of tickets requested" in the paragraph with id="ticketQty".

- Otherwise (meaning quantity is valid) perform the following steps:

  Use if statement to check that ticketPrice is >0 and in this case
  Call the function that calculates amount due for the tickets and store the returned value in ticketCost.
  Produce the following output in the relevant paragraphs:
  "Seating area X" in the paragraph with id="seatingArea" (replace X with the seating area category)
  "Price per ticket $999.99" in the paragraph with id="pricePerTicket" (replace 999.99 with the ticket price formatted to 2 decimal places)
  "Number of tickets ordered 9" in the paragraph with id="ticketQty" (replace 9 with the number of tickets)
  "Amount due inclusive of $7 processing fee is $999.99" in the paragraph with id="amountDue" (replace 999.99 with the amount due formatted to 2 decimal places)

Test and troubleshoot your program as needed.

**Screen Capture** the **HTML page** displayed in your browser and paste into **W03P.DOCX.** You need at least 7 screenshot sets:

- Valid seating category, invalid quantity
- Invalid seating category, valid quantity
- Both seating category and quantity are invalid
- All inputs are valid with EVERY seating category tested (at least 4 screenshots, mix of uppercase and lowercase entries)

**Copy and Paste** the **HTML code** and **JavaScript code** (or screenshot of your code) from your code editor into **W03P.DOCX**

## Task 7. (CREDIT Submission Task)

Make a copy of **w3P2.html** file and rename it as **w3C1.html**

- Change the title to "Ticket Sale with Credit Card surcharge", description "Credit level task" and update other meta tags as applicable including **your student name and ID.**
- Add a script element which specified a file named **w3C.js** within the **src** attribute.
- Inside the article section:
  between the paragraph with id="ticketQty" and the paragraph with id="amountDue" insert 2 more paragraphs: with id="beforeSurcharge" and with id="surcharge"

Make a copy of **w3P2.js** and rename it as **w3C.js**

The company accepts credit cards only, therefore we need to incorporate a business rule into our code: If a customer pays with American Express card, the surcharge rate is 2%, for any other card (which is Visa/Mastercard) the surcharge rate is 1%.

Create a function that takes 1 parameter – card type represented as 1 for American Express or 2 for Visa or Mastercard, and returns **surcharge rate** (surcharge rate is in %, surcharge amount is in dollars). In case of entered value not being 1 or 2, the returned value should be -1. Note, rules for discount rate could change so it is better to keep them as a separate function.

Create a function that takes 2 parameters – currentValue and rate and calculates and returns surcharge amount in $$.

Modify init():

- At the top of init create additional variables surchargeRate, surcharge, finalCost.
- Inside the if statement checking that ticketPrice is >0, **after** the statement where you calculate cost of tickets by calling the relevant function
  - Use **prompt** to get user enter 1 for American Express or 2 for Visa or Mastercard
  - Call your function that determines surcharge rate and store the returned value in *surchargeRate*.
  - If *surchargeRate* is -1, display the error message. Otherwise,
    - call the function that calculates and returns surcharge amount, store returned surcharge amount in the variable *surcharge*.
    - Now calculate final cost to include surcharge.
    - Add the following to output using document.getElementById:
      - Display “Gross Amount: $9999.99” in the paragraph with id="beforeSurcharge", where 9999.99 is ticketCost
      - Display “Card fee: $9.99” in the paragraph with id="surcharge", where 9.99 is surcharge amount in $$
      - Display “Amount due: $9999.99” in the paragraph with id="amountDue", where 9999.99 is *finalCost*

Test and troubleshoot your program as needed.

**Screen Capture** the **HTML page** displayed in your browser and paste into **W03C.DOCX.** You need to provide at least 5 screenshot sets (values should be different from pass task tests):

- Valid seating area category, invalid quantity
- Invalid seating area category, valid quantity
- Both inputs are invalid
- All inputs are valid but credit card type is invalid
- All inputs are valid and payment is by American Express
- All inputs are valid and payment is by Visa

**Copy and Paste** the **HTML code** and **JavaScript code** (or screenshot of your code) from your code editor into **W03C.DOCX**