ADVANCED PROGRAMMING

UNIT 20

PRACTICAL WORKSHOP

SESSION 02

Part 1

– Review the previous session practical work

– Introduce today's problem

– Review problem solving techniques

– Discuss and explore the problem

Part 2

– JavaScript Activity

LAST TIME AROUND

HOW DID YOU PROGRESS?
PRACTICAL REVIEW

Reflect & consider the following:

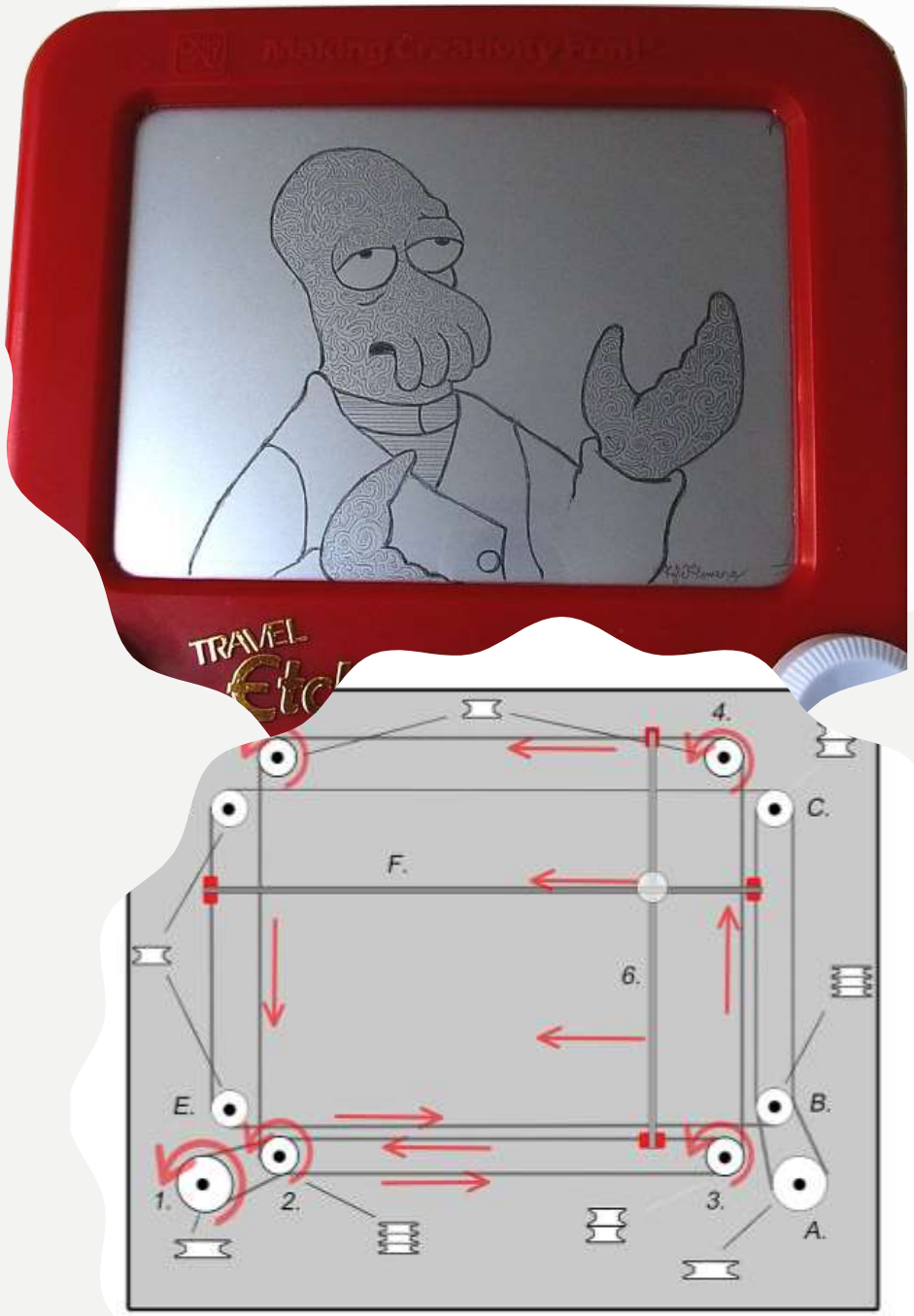
1. Did you complete each Tutorial? When, where?
2. What bits were easy? Why?
3. Where were the challenges?
4. How did you deal with the more complex bits?
5. What were the more complex bit?
6. Were you able to use any previous/past experience?
7. How will you approach this session's tasks?
8. What will you you differently?

Think about your answers and give your overall effort (for last week) a rating out of 4

You may be asked to explain and/or compare it to your later reviews

TODAY'S PROBLEM STATEMENT

Design, test, code and debug a JavaScript program that provides an Etch-A-Sketch style experience. Your program must allow a user to draw a picture by moving a stylus along the X and Y axis; as the stylus moves it leaves a single grey line. Thicker lines can be produced by drawing additional or adjacent lines. The stylus must be controlled using buttons for: UP, DOWN, LEFT and RIGHT.



HOW TO SOLVE THE PROBLEM

USE DECOMPOSITION & ABSTRACTION

PROBLEM SOLVING (1/2) – BUILD AN ALGORITHM

Decomposition:

- Explore, discuss and break the problem in to a number of specific tasks.
- Turn each task into a series of simple steps.
- Make simple notes about each step.
- Avoid "human" level thinking (e.g. assumptions or high-level statements).
- Review your steps by walking through them one and a time.
- Correct or refine any identified issues.
- Create a flowchart based on the steps.
- Test your flowchart's logic with "test data".

Abstraction:

- Ignore any aspects of the problem or specific tasks that don't need to be solved yet (or don't need to be solved by you).
- Use this process to create a list of things you may need to investigate further.
- Keep your solution simple, clear but specific.
- Try using pseudocode to help workout how to complete any functions or processes you identified in your flowchart (or that need to be programmed).

PROBLEM SOLVING (2/2) – PROGRAMMING SOLUTIONS

1. Don't build the entire solution in one instance
2. Focus on the easy bits first.
3. Research and create small prototypes to test and improve your understanding of any functions or behaviors needed for your solution.
4. Once you're happy a bit is done move to the next.
5. Save and test your code each time you make a change.

Helpful Suggestions

- Keep copies of all your work (a version number to early prototypes, e.g. webpainter_01.html).
- Use your previous programs as working examples on how you solved a previous problem.

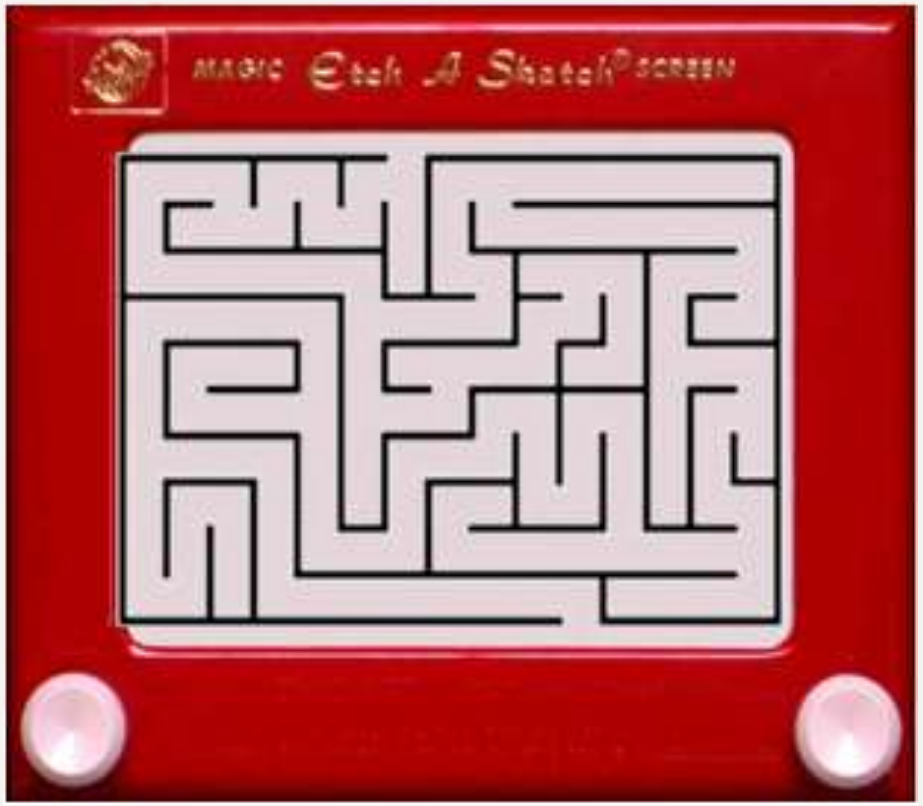
TODAY'S PROBLEM: ETCH-A-SKETCH

USE A STYLUS TO DRAW LINES ALONG AN X & Y AXIS (UP, DOWN, LEFT, RIGHT)

Example Decomposition (break down the problem)

Identify the basic components or tasks

1. Create a canvas that the user can "etch" (draw) on.
2. Create a stylus that the user can control.
3. Allow the user to move the stylus left.
4. Allow the user to move the stylus right.
5. Allow the user to move the stylus up.
6. Allow the user to move the stylus down.
7. Anything else?



Things to consider:

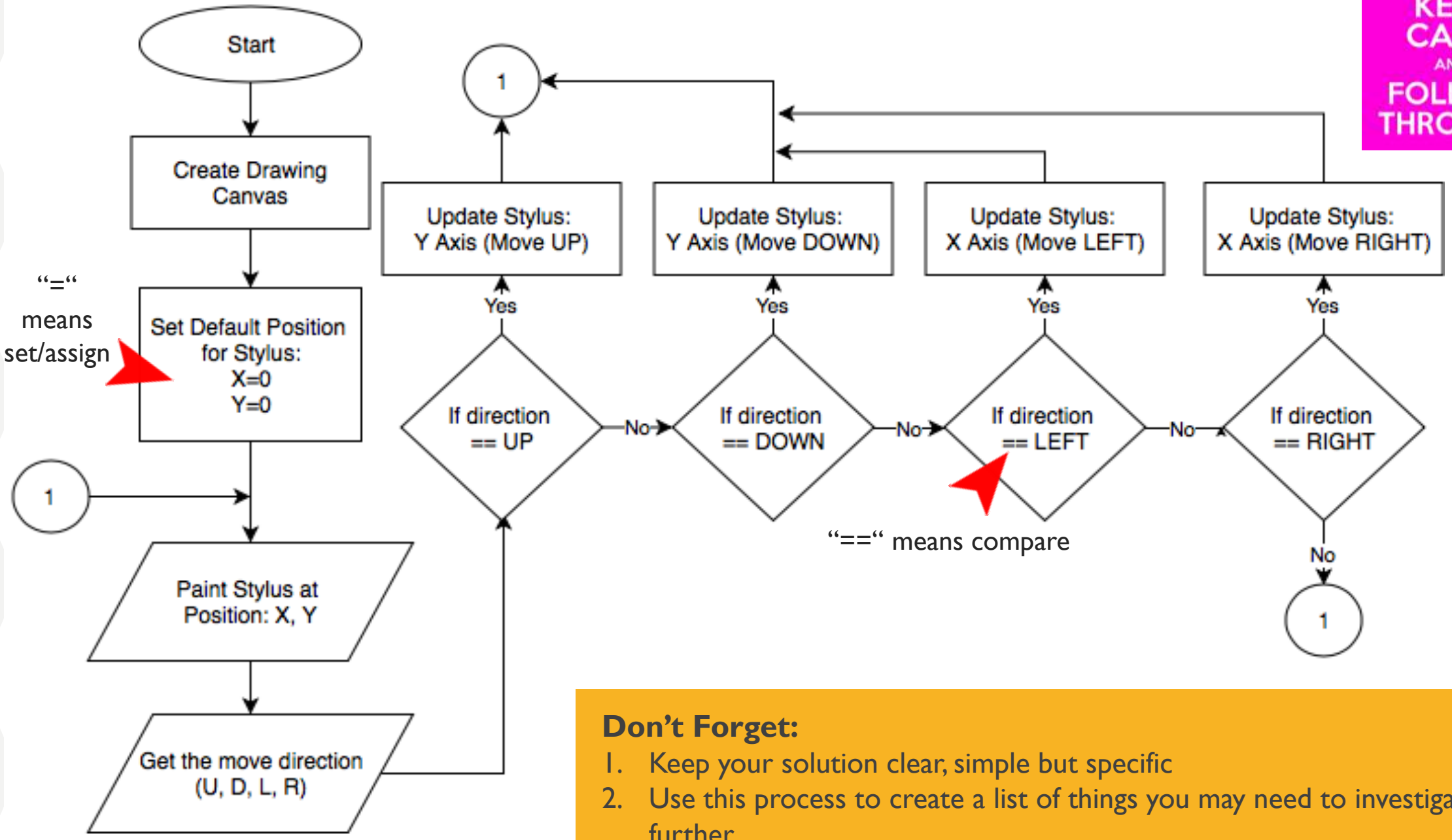
- How we are using decomposition & abstraction.
- What information do we need.
- When do we need it.
- What will we do with it.
- Next steps

FLOWCHART: ETCH-A-SKETCH (10/15MINS)

Design, test, code and debug a JavaScript program that provides an Etch-A-Sketch style experience. Your program must allow a user to draw a picture by moving a stylus along the X and Y axis; as the stylus moves it leaves a single grey line. Thicker lines can be produced by drawing additional or adjacent lines. The stylus must be controlled using buttons for: UP, DOWN, LEFT and RIGHT.

1. **As an individual or in groups (of 2/3) review the problem and consider the following:**
   - *What information does the program need?*
   - *What information can the program provide?*
   - *When and where will that information be provided or calculated?*
     The purpose of this stage is to help you start to build a logical approach to problem solving. Start with simple questions like, "what has to happen first", "okay, now what?", "then what?", etc – repeat this process until you feel you have broken down all the steps needed to solve this problem.

2. **Got a possible idea for your algorithm and the steps and the order needed?**
   - *Based on your notes (or from memory) try to construct a flowchart (either on paper or via the DrawIO site)*
   - *Test your flowchart by carefully walking through each step, use this time to try to improve or refine your solution so it works and is as simple as possible*



Don't Forget:

1. Keep your solution clear, simple but specific
2. Use this process to create a list of things you may need to investigate further.

FURTHER INVESTIGATION & RESEARCH?

- **Creating a "Canvas"**
  - e.g. something to draw/paint on
  - Conveniently HTML5 uses the tag <canvas> </canvas> to provide that exact feature
- **Positioning and painting a "Stylus"**
  - e.g. this could simply be a small square, a bit like a mouse pointer
  - To keep things simple could create a function that draws a small square at an XY position
- **Moving the "Stylus"**
  - e.g. we need something that can update the position of the stylus
  - Technically do we need to move the stylus? What would the effect be if we just repaint the stylus?
- **Detecting when a button is "Clicked"**
  - e.g. we have already code some working code for "buttons" so can reuse and develop that?

Although these issues are discussed in the comments of each tutorial task - you will be expected to do something similar (independently) when it comes to completing your coursework challenges!

PRACTICAL WORK

01

Download and try to complete the following tutorials in order:

- etchasketch_tutorial_01.html
- etchasketch_tutorial_02.html
- etchasketch_tutorial_03.html
- etchasketch_tutorial_04.html

02

Open in your preferred editor then read and review the code. Each tutorial contains a number of tasks that are specifically designed to get progressively more challenging.

03

This weeks challenge is to complete each of the tutorials by the start of your next session; we will be building on that experience so try not to miss these opportunities

TUTORIAL TASKS

Each tutorial sample is:

- A progressively more evolved prototype solution for the overall problem.
- Designed to help isolate and explore different aspects, instructions, functions and possible problem resolution.

Each tutorial sample contains:

- Details on what is happening and why.
- Fully commented code (mostly).
- Information on the tasks that need to be completed.
- More complex challenges, some may even contain errors (don't forget to enable the browser console to see debug and diagnostic information).

Completing each tutorial in order will:

- Provide you a graduated set of challenges.
- Help build your experience.
- Set a final challenge for you to complete before the next session.

QUICK POINT

AS YOU PROGRAM

Reflect and use your flowchart to:

1. Help visualise which part of the program you are working on.

2. Check which parts you have already completed.

3. Better understand the role and use of flowcharts in programming.

4. Try to identify any unnecessary or duplicated code.

5. Review additional functions coded that were not on the flowchart.

A QUICK TEST

Once you've got the resources, using the first tutorial - see if you can change the name shown of the browser tab from:

"**Tutorial XXX**"

to

"**Etch A Sketch 01**"

Once done (and tested), review the comments and tasks defined in the source code of each tutorial.

Don't forget:

- Organise your workspace (editor & browser) and save your changes and refresh your browser to test.