CODELAB II

ASSESSMENT 2: Data Driven Application

Module Coordinator: Jake Hobbs

Marker: Jake Hobbs

Contribution towards overall module mark	70%
Date set	December 3, 2018
Marked work returned by	February 22, 2018
DEADLINE DATE	January 31, 2019 - 23:59

Assessment 2: Data Driven Application

The Brief

For this assignment you are asked to develop an application that interrogates tweets. You have two options for completing this assignment. For both options the application should be designed to use functions, and pass arguments between these as appropriate. Where possible the application should also implement object oriented programming techniques.

Option 1:
Utilise the 'sampleTweets' file that contains the downloaded tweets from the top 20 users on Twitter. Your program must provide a minimum of 10 sample queries that should include

- Count the total number of tweets in the data set
- Count the number of tweets that mention the word money.
- Count the number of tweets that mention politics.
- Print to the screen any tweets mentioning the word "Paris"
- Print to the screen any tweets mentioning the word "DreamWorks"
- Print to the screen any tweets mentioning the word "Uber".
- Plus a minimum of 4 queries of your own.

The final application should be delivered via the console. **In addition** you should provide a mock GUI that demonstrates how you imagine your user interface would look. This GUI should be built using openFrameworks.

Option 2:

The application should query realtime tweets directly from twitter using the ofxTwitter addon for the openFrameworks C++ toolkit. Your application should provide a minimum of 10 different queries via a menu. Sample queries may include but are not limited to:

- Display tweets from a certain date *(limited to last 7 days with standard search API)*
- Display series of tweets from a certain hashtag
- Display series of tweets mentioning a specific user
- Display tweets from a certain location
- Display tweets mentioning “Bath Spa University” from within 10 miles of Bath
- Display most popular tweets mentioning “Donald Trump”
- Allow free search queries

You should aim to include variety in your search queries, by refining the searches with the parameters available via the twitter search API and search filters. Exploration of the API, the sample files included in the Twitter Addon and the the header files of the ofxTwitter addon will aid your in enhancing your app.

The header files can be found in:
openFrameworks/addons/ofxTwitter/libs/ofxTwitter/include/ofx/Twitter

The final application should function via a GUI window

Deliverables

The deliverables for the Data Driven Application are as follows:

The Application

The C++ code (e.g. main.cpp file) and any supplementary assets (images, sounds, fonts) required to run your data driven application.

For Option 1 this will be the the code files (.cpp) you have used to develop your solution. You should also submit the **src** and **data** (located inside the bin) folders from your openFrameworks mock GUI project.

For Option 2 this will be the **src** and **data** (located inside the bin) folders of your openFrameworks project.

For both options all the files you are required to submit should be tracked by Github desktop, so all you should need to do is commit and push your code back to your unique github repository for the assignment. However, If you are unsure - just ask!

The Development Document

Your data driven application must be accompanied by a Development Document of a minimum of 1000 words (there is no maximum word count). This development document should consist of the following elements:

- *Abstract:* A short description of what you have been asked to build, including any notes needed to run the program. For Option 2 this should include a list of any additional openframeworks addons you may have used beyond those required by ofxTwitter. Please also ensure you provide links to where these additional addons can be downloaded from.

- *Project Plan:* Project plan specifying key milestones (e.g. via a gantt chart)
- *Evidence of design:* This may include but is not limited to: specification list of your programs requirements, Flowchart, Psuedo-Code, Wireframes, UML Data Structure Diagram
- *Technical Description:* A technical breakdown of how your application operates. This should use accurate terminology to describe the your approach to achieve the final outcome. You may wish to refer to your design work (pseudo-code or flowcharts) in this section.
- *Testing:* Test plan, test data and evidence of testing (e.g. screenshots and or videos).
- *Critical Reflection:* An open and detailed evaluation of your application that notes what is compelling about the work, what could be improved, and what you need to learn to make these improvements.
- *Appendix:* A copy of your code should be included in an appendix at the end of your documentation.

Submission

Please follow the submission instructions below. Work that is submitted incorrectly may not be accepted or could incur a points penalty.

Before submitting have you…

- Spell-checked and grammar-checked your work? Please make an appointment with the Writing and Learning Centre or speak to your tutor if you are experiencing challenges in this area.
- Formatted your written work to the specification below?
- Referenced all sources of information accurately? Please refer to www.citethemrightonline.com for guidance.

Your development document must be submitted via Turnitin with the coversheet included as the first page. Please adhere to the following method:

- Check your Data Driven Application is working as expected.
- Commit and push your final version to your unique Github repository for this assignment. Ensure you push all the files required to run your application. Only code committed to Github will be marked
- Ensure your github profile bio contains your student number
- Copy the link for your unique github repository
- Paste the URL to the assessment cover sheet provided on Minerva.
- Include the cover sheet as first page of your development document and save this as a Word document.
- Log into Minerva, go to the Assessment tab and submit your finalised document via the appropriate Turnitin Link.

Format

All essays and reports must conform to university styling and submission guidelines.

They must:

- Be word-processed using a conventional font and size (e.g. Times New Roman, 11 or 12) and 1.5 or double line spaced on single-sided paper.
- Contain appropriate in-text citation that supplies an accurate list of references.
- Be accurate in referencing. See Bath Spa guidelines
- Be accurate in spelling and paragraphing.

Marking Criteria

Assignment 2: Data Driven Application will be marked against the following criteria:

1. System Design
2. Evidence of Testing
3. Documentation
4. Technical Implementation

Please pay particular attention to the differences in Technical implementation between Option 1 and Option 2. Marks for technical implementation are capped on option 1, however all other marketing criteria are unaffected by the option you take. Therefore a first overall is still possible with option 1 despite the limits placed on the technical implementation marking criteria.

Criteria	Weighting		Marks
System Design	15%	A very limited design that may not be implementable.	0 - 19 (Low Fail)
		A poor system design that omits key features. The application may not be fit for purpose.	20 - 39 (Fail)
		A basic design that meets the minimum requirements stated on the brief.	40 - 49 (Third)
		A fair design that meets the minimum requirements well but does not offer additional features.	50 - 59 (2:2)
		A good design that meets the minimum requirements well and presents some additional features. May include original search queries beyond those suggested in the brief.	60 - 69 (2:1)
		Very good system design that provides several additional features. Includes original	70 - 79 (First)

		search queries beyond those suggested in the brief.	
		Excellent system design that includes many original additional features and implements techniques beyond those taught in class.	80 - 89 (High First)
		Beyond expectations for this level of study.	90 - 100 (Outstanding)
Evidence of testing	20%	No evidence of testing.	0 - 19 (Low Fail)
		Limited evidence of testing.	20 - 39 (Fail)
		Basic test plan and data. No evidence of independent user testing.	40 - 49 (Third)
		A fair test plan and data. Evidence of independent user testing and how this affected the final solution is lacking.	50 - 59 (2:2)
		Good test plan and data with some evidence of independent user testing. May lack detail on how testing affected the final solution.	60 - 69 (2:1)
		Comprehensive test plan and test data with evidence of independent user testing and how this affected final solution.	70 - 79 (First)
		Highly comprehensive test plan and test data, with evidence of rigorous independent user testing how this affected final solution.	80 - 89 (High First)
		Beyond expectations for this level of study.	90 - 100 (Outstanding)
Documentation	25%	Very limited documentation that demonstrates little or no understanding of the design and build process. Critical reflection is missing or inadequate. Structure is unacceptable.	0 - 19 (Low Fail)
		A poor attempt that does not meet the requirements of the development document.	20 - 39 (Fail)

		May be missing key sections or badly structured. Inadequate technical description and/or critical reflection.	
		Basic documentation that offers only minimal information about the design direction and technical implementation. Correct use of technical terminology is lacking and critical reflection is limited in depth. Structure is acceptable	40 - 49 (Third)
		A fair attempt that provides only key information about the design and technical implementation. May be limited in the clarity of the technical description and depth of the critical reflection. Document structure is acceptable.	50 - 59 (2:2)
		A good to very good development document that provides a clear overview of the design direction and a sound description of the technical implementation. Critical reflection is adequate, but may be limited in depth. Document structure has only minor issues.	60 - 69 (2:1)
		Very good overview of the that provides detailed description of the concept design and technical implementation. Structure is without error. Critical reflection is well considered.	70 - 79 (First)
		An excellent overview that provides a high level of insight into the design and technical implementation. Reflection is highly critical and detailed. Structure is without error.	80 - 89 (High First)
		Beyond expectations for this level of study.	90 - 100 (Outstanding)

Criteria	Option 1	Option 2	Marks
Technical Implementation 40%	A very limited implementation that demonstrates little evidence of programming skill.	A very limited implementation that demonstrates little evidence of programming skill.	0 - 19 (Low Fail)
	A poor implementation that demonstrates a limited understanding of the brief, and may contain significant errors.	A poor implementation that demonstrates a limited understanding of the brief, and may contain significant errors.	20 - 39 (Fail)
	A fair implementation using the pre-supplied twitter data. Queries may contain errors or missing content. Mock GUI has not been implemented Minimal commenting in the code. Only a minimal use of functions	A poor implementation that attempts to connect to twitter, but does not function correctly. Code does not go beyond the examples available. Minimal commenting in the code.	40 - 49 (Third)
	A good implementation using the pre-supplied twitter data. All queries function as expected. Basic mock GUI has been implemented, but has room for refinement Code has some commenting but this is limited in clarity Programme uses functions appropriately but there is scope for improved structure and logic (e.g. greater use of parameters).	A basic implementation that connects to the live twitter service. Code makes minimal effort to go beyond the examples available Output either remains in console or GUI implementation is ill-considered Code has some commenting but this is limited in clarity	50 - 59 (2:2)
	A very good implementation using pre-supplied twitter data. All queries function as expected, including additional queries with originality Program is highly structured with functions that pass parameters appropriately in order to improve	A basic - good implementation that connects to the live twitter service. There is evidence of some attempts to go beyond the examples available Code may lack structure / efficiency	60 - 69 (2:1)

	efficiency. Mock GUI has been implemented and the design is well considered. Coding Conventions have been observed (comments, indentation, camelCase etc), with only minor errors present	Output delivered via a GUI, which functions but lacks attention to details Coding Conventions have been observed (comments, indentation, camelCase etc), with only minor errors present	
	N/A	A good implementation that connects to live Twitter services. Program queries function without error and demonstrate exploration of the twitter API beyond the examples available. GUI implementation is well considered with high attention to detail Code is commented and structured to a high standard and coding conventions have been well observed. Makes good use of classes for code reuse and efficiency	70 - 79 (First)
	N/A	An excellent implementation that deploys techniques well beyond the scope of those demonstrated in class. Programme successfully integrates with live Twitter service with the user given access to multiple potential queries that demonstrates a range of data available via the Twitter API Commenting is detailed and coding. conventions have been observed with precision. Makes excellent use of classes for code reuse and efficiency	80 - 89 (High First)
	N/A	Beyond Expectations for this	90-100

		level of study	(Outstanding)

Intended Learning Outcomes (ILOs)

ILO	Assessed
The implementation and testing of a prototype system that is driven by object-oriented programming techniques.	✓
Application of the iterative design cycle of prototyping, testing, analysing and refinement.	✓
A recognition of personal knowledge limits, which is addressed through the identification of learning opportunities.	✓
An ability to critically review the key features and challenges of developing software for an end user, and evaluate their relevance to the field of creative computing.	

Mark penalties may be applied to late submissions without prior approval of extension. Please ensure that you prepare and submit your work in good time to allow for any issues that may arise.