Redacted

Part 2

Redacted is an easy way to simplify documentation for a program. By utilising the power of OS X i have managed to bring forward a revolution is software development that will change the way that developers look at documentation. The choice of using swift was simple: it was an easy to develop for, widely accepted efficient language that offered the tools that i would need to develop something as complex as redacted. under the hood OS X provides the efficacy and the user base that this program deserves. The use of swift also brings a future version being available for mobile devise for those without OS X but has IOS.
The program Does two main things; produce a stunningly designed data dictionary that includes descriptions of variables if they've been given, and construct a reference manual that will be able to inform all of your users of the functions.

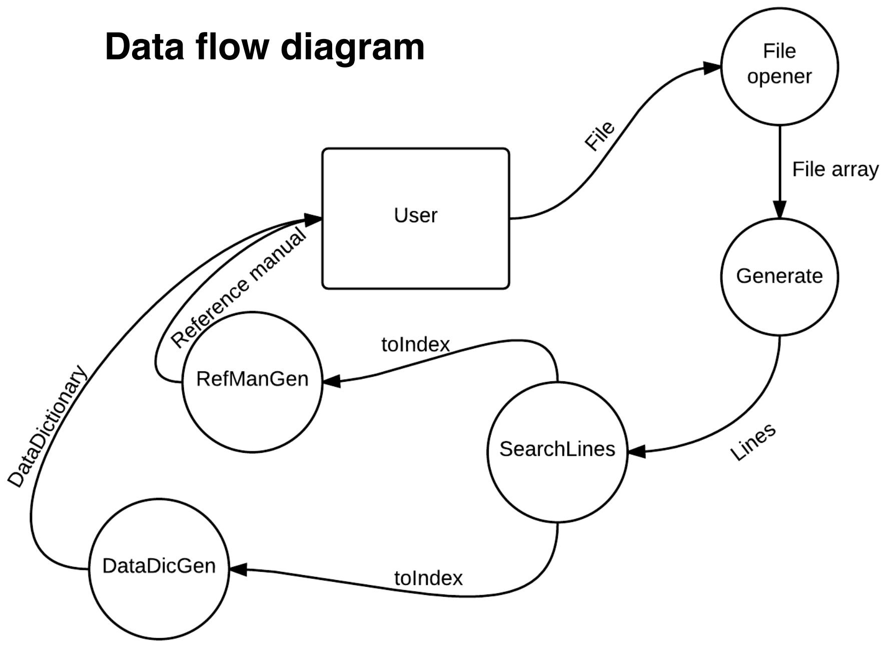
System level test data

Data	Expected output
Click open	places path of file in program and detects language
Click Generate	scans file and outputs a data dictionary and reference manual
Function in file	function is found and placed in outputs
description after function name	Found and placed after function in outputs
File empty	Returns visual exception about emptiness
Unknown language imported	Returns visual exception about emptiness
nothing to be found	Returns visual exception about lack of things to be found

Module level test data

Module	**Input**	**Output**
LinesToArray	File	Lines in an array
SearchLines	Lines array	array of the index of functions
RefManGen	Array of lines with functions	Reference manual

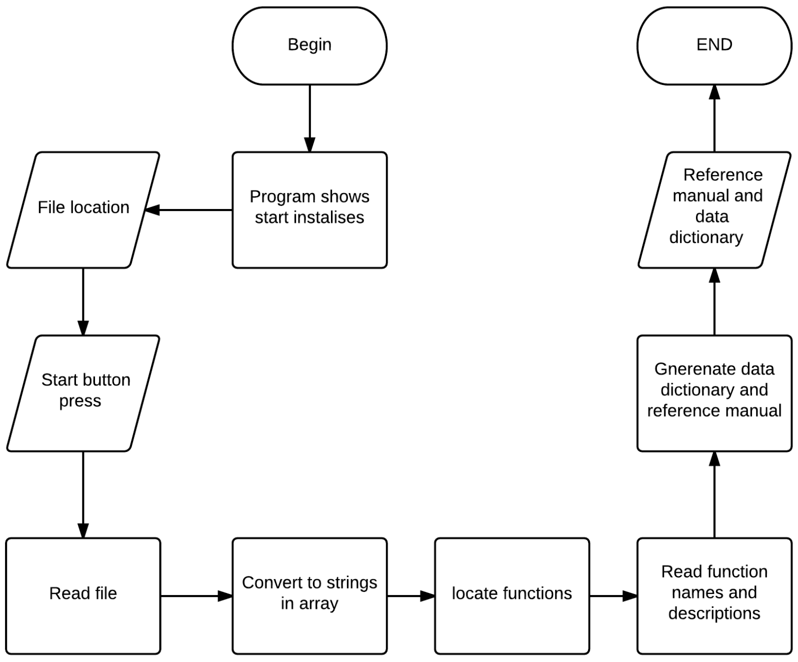
Data flow diagram



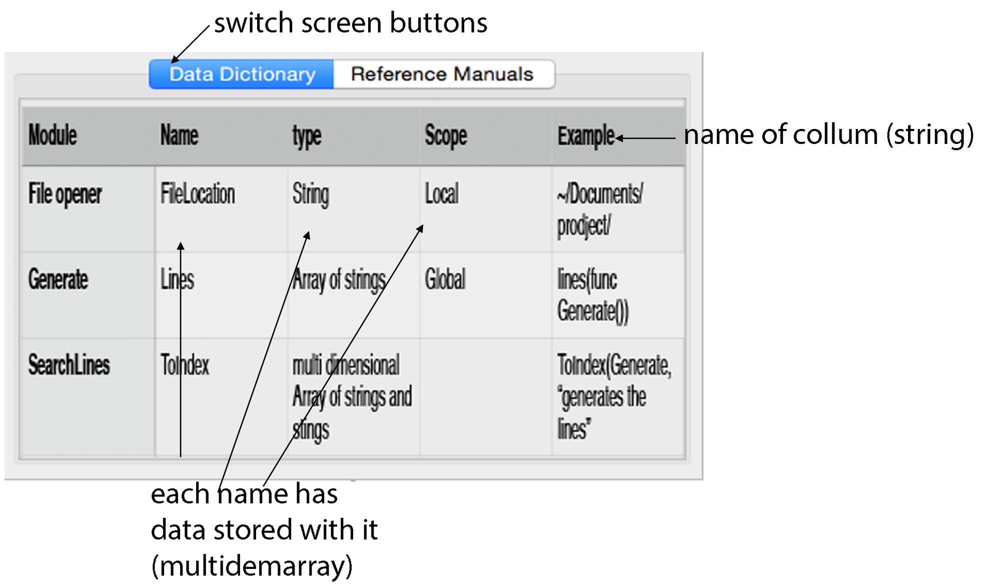
Data Dictionary

Module	Name	type	Scope	Example
File opener	FileLocation	String	Local	~/Documents/ prodject/
Generate	Lines	Array of strings	Local	lines(func Generate())
SearchLines	ToIndex	multi dimensional Array of strings and stings	Local	ToIndex(Generate, “generates the lines”
dataDicGen	DataDictionary	multi dimensional Array of strings and stings	Global	dataDicGen(1,1) = FileLocation, “opens file”
RefMangen	RefMan	multi dimensional Array of strings and stings	Global	refMan(1,1) = searchLines, “searches lines”

System Flow chart


Begin
Program shows start instalises
File location
Start button press
Read file
Convert to strings in array
locate functions
Read function names and descriptions
Gnrenate data dictionary and reference manual
Reference manual and data dictionary
END

Screen design



Module	Name	type	Scope	Example
File opener	FileLocation	String	Local	~/Documents/ prodject/
Generate	Lines	Array of strings	Global	lines(func Generate())
SearchLines	ToIndex	muli dimensional Array of strings and stings		ToIndex(Generate, "generates the lines"

Pseudo code

```
Begin LinesToArray(file)
    REM turns each line of the file into a string in an array
    File = File.readlines()
    For i = 1 to file.length:
        Lines(i) = file(i)
    NEXT I
    RETURN Lines()
END


Begin searchLines(lines)
    REM searches all the lines for certain words1
    if lines(i) contains "func"
        toIndex(index) = i
        index + 1
    ELSE
        next i
    Return toIndex()
END
```