Livingston - hmwk 8/4

@Synthesize works with the compiler to generate the getter/setter methods for the properties. It generates the methods based on property descriptions. After the xcode 4.4 version, the @synthesize is provided by default

@Synchronize helps to prevent memory leaks. There are a few tools available to do this: Memory barriers, and locks. The use of @synchroize works to manage how multiple threads are accessing memory so that they don’t affect the original piece code, thus resulting in crashes, or unexpected changes in code output. Memory barriers are used to make sure memory operations are done in the correct order, while locks are used to protect critical sections of code

@Dynamic tells the compiler that the getter/setter methods have been implemented. It also tells the compiler not to create the accessor methods because the user will provide that implementation in the future. @dynamic is also used to delegate the responsibility of accessors implementing sections of code

iVars are instance variables used within functions. This makes it harder for memory leaks and unexpected changes because they are used inside the functions. eVars are made public so that all functions have access to them. iVars aren’t needed as much anymore after the introduction of @property , and @synchronization. Because iVars are declared inside of functions, it prevents the possibility of encapsulation or data hiding. eVars, on the other hand remain in memory even after it has been returned by the function

Getter in Objective C, get called every time you access from a property. Setters get called every time a value is changed. Accessing an iVar through a getter/setter uses an objective-c method call.

Method Swizzling, also known as monkey patching, is a process of replacing functionality of a section of code before its been called by the compiler. It can also add functionality to custom code before the original is called. Swizzling can affect all functionality, including contrtollers owned by 3rd-party frameworks. Swizzling works everytime an @objc is called, and requires dynamic dispatch