|  |  |
| --- | --- |
|  | **Accessor:** return value of the object  **Mutator:** assign a value to the object  **Static:** share data among all members  **Executable Process:** convert java code to bytecode (.class) and load class into JVM **Assertion**: check for property exists in system, throws error if there is an error, help check code at runtime (default: not used), only use for testing purpose **Sub-Typing**: generalization (going from a specific class to general class) and specialization (going from general class to specific class) **Covariant**: the ordering of component type is in order(asc) **Contravariant**: the ordering is reserved(des) **Invariant**: neither covariant or contravariant  **Function Pointer**: hold the address of another function **Immutability**: create an object to make any necessary change before passing to the target value |
|  | **Logger**: use to diagnose problems when there is no debugger (more likely to be a print statement but contain code checking, zone determined by coder) **GUI Event Handling**: event source object (fire an action event), an event object, event handler object(capable of processing event) **Socket**: abstraction which allows communication between 2 programs, opened via address and port. **Collection**: same as a container in C++, there is List<>, Set<> and Queue<>. They all belongs to Collection class (which collection type can use a built-in function such as add(E e), remove(Object o), size(), isEmpty()). The collection class also belongs to Iterable class which contain the get for iterator(), forEach(class<> action) **Confinement**: disallow shared data.  **Locking**: safety and speed, synchronized data for safety purpose |
|  | **Observable**: allow developers to be prompts whenever there is a value changing(either removing, assigning new one or set to some default). This is mostly the same as debugging.  **Event Handling**: dispatcher takes events and assignment, correct dispatchers, to handle the correct event type (similar to Switch (...) case…:). This allows the user to trigger any event at any time and the program will still be able to handle correctly. **JDBC Technology**: connect to database → create a statement and execute query → check the result → close connection, JDBC is created so that the application doesn’t care about the database format but ensure that the code should work across all type(ensure to specify driver of the database). JDBC helps the developer to access database SQL using SQL statement on the Java application platform |



