Design patterns for mobile development.

1st Jesus Antonio Panama Arellano
Multiplatform Software Development
Universidad Tecnologica de Tijuana
Tijuana, Mexico
0322103781@ut-tijuana.edu.mx

***Abstract*—Design patterns for mobile development offer reusable solutions to common mobile application design challenges. These proven practices and methodologies not only speed up the development process, but also improve user experience and customer satisfaction. They fall into three main categories: authoring patterns, structural patterns, and behavioural patterns. Creation patterns, such as Abstract Factory and Singleton, address object creation. Structural patterns, such as Adapter and Composite, facilitate the composition of classes and object structures. While behavioural patterns, such as Observer and Strategy, deal with communication between objects and variation of class behaviour. These patterns provide a solid framework for the efficient and effective design of mobile applications.**

***Index Terms*—Development, Mobile, Patterns**

I. Introduction

Design patterns are fundamental tools in software development, and mobile is no exception. These patterns offer proven, reusable solutions to common challenges developers face when designing mobile applications. From creating objects to managing communication between them, design patterns provide a solid framework for efficient and effective mobile application development.

II. Development patterns

They refer to general, reusable solutions that address common design problems and challenges encountered when designing mobile applications. These design patterns reference proven best practices and methodologies that help developers create efficient, aesthetically appealing and easy-to-use mobile applications. Your use of mobile design patterns not only helps speed up the development process, but also contributes to a better user experience and increased customer satisfaction.

- Creation patterns: They are a collection of ideas for using objects and classes to build larger structures while maintaining flexibility and efficiency. They allow connecting components in an application, and are beneficial in more complex applications.
  - Abstract factory In this pattern, an interface creates sets or families of related objects without specifying the class name.
  - Builder Patterns Allows different types and representations of an object to be produced using the same builder code. It is used for the step-by-step creation of a complex object by combining simple objects.
  - Factory method provides an interface for creating objects in a superclass, but does not allow subclasses to alter the type of objects that are created.
- Structural patterns: They facilitate efficient solutions and standards with respect to class compositions and object structures. The concept of inheritance is used to compose interfaces and define ways of composing objects to obtain new functionalities.
  - Adapter.
  - Bridge
  - Composite
- Behaviour patterns: Deals with communication between class objects. They are used to detect the presence of communication patterns already present and can manipulate these patterns.
  - Chain of responsibility Avoids coupling the sender of a request to its receiver by giving more than one object the possibility to respond to a request.
  - Command Converts a request into an independent object containing all the information about the request. This transformation allows parameterizing methods with different requests.
  - Interpreter Used to evaluate the language or expression by creating an interface that indicates the context for the interpretation.
  - Mediator Provides easy communication through its class that allows communication between various classes.
  - Observer Allows defining a subscription mechanism to notify several objects about any event that happens to the object being observed.
  - Strategy Allows defining a family of algorithms, put each of them in a separate class and make their objects interchangeable.

References

[1] J. Cáceres, “Patrones de diseño”, Rev. Educ. Distancia, 2009.