Contents

[1. Introduction 2](#_Toc29909814)

[1.1 Software Programming 2](#_Toc29909815)

[1.2 Types of software programming 2](#_Toc29909816)

[1.3 Software Engineering 3](#_Toc29909817)

[1.4 Software Paradigms 3](#_Toc29909818)

[1.5 Characteristics of properly software program 4](#_Toc29909819)

[2. Software Development Life Cycle 5](#_Toc29909820)

[2.1 Agile Software Development Lifecycle 7](#_Toc29909821)

[2.2 Main Stages in Agile challenge: 8](#_Toc29909822)

[2.3 Advantage and Disadvantage of Agile Methodology: 9](#_Toc29909823)

[3. Version Control 10](#_Toc29909824)

[3.1 Why Use a Version Control System? 11](#_Toc29909825)

[4. Design pattern 13](#_Toc29909826)

[4.1 Elements of a Pattern 14](#_Toc29909827)

[4.2 Factory Design Pattern 15](#_Toc29909828)

[5. Unit Testing 15](#_Toc29909829)

[5.1 Advantage of unit testing 16](#_Toc29909830)

[6. File Handling in C# 19](#_Toc29909831)

[7. Handling and raising events 20](#_Toc29909832)

[7.1 Events 20](#_Toc29909833)

[7.2 Delegates 21](#_Toc29909834)

[7.3 Event handlers 22](#_Toc29909835)

[8. Reference 23](#_Toc29909836)

# Introduction

Think about a number of the distinct approaches that people use computers. In school, college students use computer systems for tasks along with writing papers, attempting to find articles, sending email, and taking part in

on-line classes. At work, human beings use computers to analyze data, make presentations, conduct business transactions, speak with clients and coworkers, manipulate computers in manufacturing facilities, and do many different things. At home, humans use computer systems for tasks which include paying bills, shopping on line, speaking with pals and family, and gambling computer games. And don’t overlook that cell phones, iPods®, BlackBerries®, car navigation systems, and many different computers are computers too. The makes use of of computers are almost endless in our ordinary lives.

Programs are commonly called software. Software is vital to a computer because it controls the whole lot the computer does. All of the software program that we use to make our computers beneficial is created through individuals operating as programmers or software developers. Programmer, or software developer, is someone with the schooling and abilities important to design, create, and check laptop programs. Computer programming is an interesting and worthwhile career. Today, you may locate programmers’ work utilized in business, medicine, government, law enforcement, agriculture, academics, entertainment, and plenty of different fields

## 1.1 Software Programming

Software programming a profession within the computer technology field that mainly offers with writing code. Read directly to get a programming and software development definition as well as a laptop software programmer job description. Schools providing Application Development degrees also can be determined in these popular choices.

## 1.2 Types of software programming

Software packages are normally categorized into the programming languages which are like minded with them. There are many styles of programming languages in existence, but underneath is a list of some famous codes and what they're used for.

JavaScript. JavaScript is generally used on websites to feature interactive elements.

SQL (Structured Query Language). SQL is a database query language that lets in websites to transfer data from massive databases.

Python. Python is a language used for a wide kind of things, from net apps to facts analysis.

Java. Java is typically utilized in video games and mobile apps, such as apps for Android devices.

C#. Comparable to Java, C# is used for Microsoft apps.

Many of these packages provide certification from the organisation that evolved them. For instance, Oracle has the Oracle Certified Associate Java Programmer (OCAJP) and the Oracle Certified Professional Java Programmer (OCPJP) certification. Certification usually involves passing an exam; getting licensed is an critical step in proving your know-how and finding employment as a pc programmer.

## 1.3 Software Engineering

software engineering is an engineering branch associated with development of software program product using well-described scientific principles, techniques and procedures. The outcome of software program engineering is a good and reliable software product.

IEEE defines software engineering as:

(1) The utility of a systematic, disciplined, quantifiable approach to the improvement, operation, and maintenance of software; this is, the utility of engineering to software program.

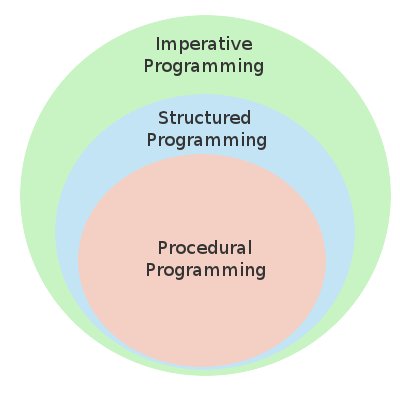
(2) The observe of techniques as inside the above statement.

Fritz Bauer, a German pc scientist, defines software engineering as:

“Software engineering is the establishment and use of sound engineering principles in an effort to obtain economically software that is dependable and work efficaciously on real computers.”

## 1.4 Software Paradigms

Software paradigms discuss with the methods and steps, which are taken even as designing the software program. There are many techniques proposed and are implemented. But, we want to see where in the software program engineering concept, those paradigms stand. These can be mixed into diverse categories, even though every of them is contained in one another:



Programming paradigm is a subset of Software design paradigm which is further a subset of Software development paradigm

## 1.5 Characteristics of properly software program

A software product may be judged via what it offers and how well it may be used. This software program must fulfill on the following grounds:

* Operational
* Transitional
* Maintenance

Well-engineered and crafted software is predicted to have the following characteristics:

**Operational**

This tells us how properly the software works in operations. It may be measured on:

* Budget
* Usability
* Efficiency
* Correctness
* Functionality
* Dependability
* Security
* Safety

**Transitional**

This factor is important whilst the software is moved from one platform to another:

* Portability
* Interoperability
* Reusability
* Adaptability

**Maintenance**

This factor briefs about how well the software program has the abilities to hold itself inside the ever-changing environment:

* Modularity
* Maintainability
* Flexibility
* Scalability

# Software Development Life Cycle

Software Development Life Cycle, SDLC for short, is a well-defined, structured series of stages in software engineering to increase the supposed software product. SDLC Activities

SDLC provides a series of steps to be accompanied to design and broaden a software product efficiently. SDLC framework includes the subsequent steps:

**Communication**

This is step one wherein the person initiates the request for a favored software product. The user contacts the carrier provider and tries to barter the terms, submits the request to the carrier providing organization in writing.

**Requirement Gathering**

This step onwards the software development team works to carry at the project. The crew holds discussions with numerous stakeholders from problem domain and tries to carry out as much information as feasible on their requirements. The necessities are contemplated and segregated into user requirements, system necessities and functional requirements. The necessities are gathered using a number of practices as given

* studying the existing or obsolete device and software,
* engaging in interviews of users and developers,
* regarding the database or
* collecting answers from the questionnaires.

**Feasibility Study**

After requirement gathering, the team comes up with a rough plan of software program process. At this step the group analyzes if a software can be designed to meet all requirements of the consumer, and if there may be any possibility of software being no more useful. It is also analyzed if the mission is financially, practically, and technologically feasible for the organization to take up. There are many algorithms available, which assist the builders to conclude the feasibility of a software program undertaking.

**System Analysis**

At this step the builders determine a roadmap of their plan and try to convey up the satisfactory software program model appropriate for the venture. System analysis consists of information of software program product limitations, learning computer related troubles or modifications to be completed in existing structures beforehand, identifying and addressing the impact of assignment on agency and personnel etc. The undertaking crew analyzes the scope of the assignment and plans the schedule and assets accordingly.

**Software Design**

Next step is to carry down complete knowledge of requirements and analysis on the table and design the software program product. The inputs from users and records amassed in requirement gathering phase are the inputs of this step. The output of this step comes inside the shape of two designs; logical design, and physical design. Engineers produce meta-statistics and statistics dictionaries, logical diagrams, facts-drift diagrams, and in some instances pseudo codes.

**Coding**

This step is likewise known as programming section. The implementation of software program design begins in terms of writing program code in the proper programming language and developing error-unfastened executable programs efficiently.

**Testing**

An estimate says that 50% of entire software program development process have to be examined. Errors may destroy the software from crucial stage to its own removal. Software trying out is achieved even as coding by means of the developers and thorough trying out is conducted by using checking out professionals at various levels of code consisting of module trying out, program checking out, product checking out, in-house trying out, and trying out the product at user’s end. Early discovery of mistakes and their treatment is the important thing to reliable software.

**Integration**

Software may additionally want to be included with the libraries, databases, and other software(s). This degree of SDLC is involved inside the integration of software program with outer international entities.

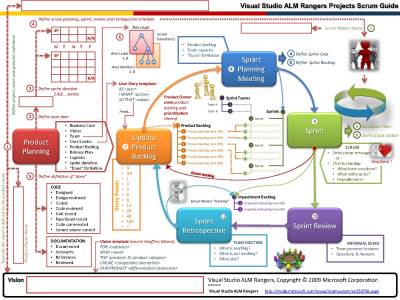
**Implementation**

This means installing the software on consumer computers. At times, software wishes post-set up configurations at consumer end. Software is tested for portability and flexibility and integration related issues are solved in the course of implementation.

**Operation and Maintenance**

This phase confirms the software operation in phrases of extra efficiency and much less errors. If required, the customers are educated on, or aided with the documentation on how to function the software program and how to preserve the software operational. The software is maintained timely by updating the code in line with the adjustments taking vicinity in consumer end surroundings or technology. This phase may additionally face challenges from hidden insects and real-global unidentified troubles.

## 2.1 Agile Software Development Lifecycle

Agile methodology is an approach to the project management which allows to reply to the unpredictability of constructing software thru incremental, iterative paintings cadences, recognized as sprints. This methodology become developed to cope with state of affairs where the waterfall model fails. The biggest downside of waterfall model is that it assumes that every requirement of the venture can be diagnosed before any design or coding occurs. This may always be relevant for the improvement of an automobile on a meeting line, in which every piece is delivered in sequential phases. However, it is able to or might not be relevant for software improvement. For example, for a BAU (Business as Usual) projects wherein the software program is already in use for a protracted time. 

waterfall version is the first-class method to implement any changes requests because quantity of uncertainly is very much less compared to growing a new product. However, for developing brand new software program, waterfall model isn't always an ideal preference as the amount of uncertainty in terms of requirement and person best expectation. It might be the case that give up product isn't precisely what person has expected because of mismatch of requirement understanding between user and developer. It might also be the case that a group may have constructed the software program it turned into requested to build, but, in the time it took to create, commercial enterprise realities have changed so dramatically that the product is irrelevant. In that scenario, an employer has spent money and time to create software that nobody wants. Agile development method offers the possibility to assess the route of a

project at some point of the development lifecycle. It does it through an iterative cycle to build and check observed by an evaluation through the person/business until they are happy with the product. Thus with the aid of focusing at the repetition of abbreviated paintings cycles as well as the functional product they yield; agile method could be described as iterative and incremental. Agile techniques should not be pressured with the Spiral technique in any case. Spiral method pressure you to plot for all of the new release inside the beginning wherein as Agile gives you the flexibility to plan simplest which you sincerely recognize and leaves the relaxation of planning for next iteration. In Spiral methodology since you have got planned for all of the generation inside the starting, the quantity of iterations is fixed. But in Agile, you could have as an awful lot new release required generating the final product as the making plans is dynamic

## 2.2 Main Stages in Agile challenge:

* Define preliminary requirement: Business will provide you with a high degree requirements to the task. It isn't always necessary to have all of the requirement finalized and clear in this level but enterprise have a good idea approximately the facilities or product they want. For example, business want a resource management product but may not be clean whether they need a desktop application or web primarily based one.
* Prepare high level use instances: Based on the preliminary commercial enterprise requirement, venture business analyst will have in addition dialogue with purchaser to have a deeper know-how to come back up with high level use cases. These use cases may be discussed with the customers to finalize the requirement at a better level.
* Prepare High Level Plan: Once the excessive level business instances are authorized and finalized, Project Manager will put together a excessive stage plan. Normally the preliminary plan has a short making plans horizon and particularly covers the plan till the first prototype completes.
* Begin Iteration: This segment is where all the groups involved in the assignment start their planning and designing phase. For example, development crew may produce Mid-Level Platform Design and Test group may also produce Master Test Plan.
* Build and Unit Test: Development take a look at construct and unit check the prototype in this stage. If it is not the first generation then they do the enhancement to the prototype of the closing generation as in line with the refines use case.
* System Test: Testing crew plays the checking out of the prototype as consistent with their check cases and test plan organized in previous stages. If it is not the primary prototype, the test team refines their check case and check plan consistent with the delicate use cases and carry out testing. Regression trying out will also be required to prove that capability tested in earlier iterations are still working properly.
* User Evaluation: Once the prototype passes the computer take a look at, representatives from the client examine the prototype against the necessities and the use cases. They additionally access if they need extra functionality to be introduced to the product as the preliminary requirement amassing stage might not have amassed the whole user requirements or were not clean at that time.
* Are greater modifications required? This isn't always a stage however a selection point inside the assignment life cycle where client decides whether or not they want more changes to the goods or not based totally on the evaluation done inside the closing stage. Prototype may work into one extra generation if customer need more modifications to the product. If they're satisfied with the product and are prepared to simply accept this as a final product, then the product may work the final spherical of device check.
* Refine Requirement /Use case: If consumer decides to make endorse modifications after the User Evaluation stage, necessities and in flip consumer instances want to be subtle as according to the change request. Development and testing group will use these adjustments use cases to refine their design and check plan.
* Final System Test: Once the client approves the prototype as the final product, a very last computer integration test is probably required to check the combination of the product with consumer environment and different application. User also might need to do a final evaluation of the product with all the different application integrated to it.
* Product Release: Once final test is done, product is ready to be released and deployed at the consumer location. Training and demo is probably required for the wider consumer group along with the person manuals.

## 2.3 Advantage and Disadvantage of Agile Methodology:

**Advantages:**

* No Detail requirement needed: You don’t want to have the entire necessities finalized to begin the improvement work. Build and Test can start as quickly as initial high level requirements are available.
* Early gain to the user/business: Following Agile methodology offers an early view to the user approximately how the very last product may look and behave. This allows them into finalizing the user requirements. It may additionally take place that the prototype delivers a number of the necessities which use may need to use and evaluate until the very last product is delivered with all of the functionalities.
* Face to face verbal exchange: Agile approach give greater emphasis on having the face to face communique among the user/patron and assignment group to make certain there may be no room left for any sort of confusion in understanding requirement and inputs.
* Less time to market: Using Agile method, very last product is added to the purchaser in least feasible time.
* Less cost to client: It saves fee for both client and supplier as assets are used
* for much less time.
* High Quality: Since consumer is worried in all of the levels of software development, means the best of final first-rate if excessive ensuing in surprisingly satisfied patron.

**Disadvantages:**

* Smaller Planning Horizon: Since Agile tasks have smaller planning horizon meaning that assignment is commenced without detailed planning, there is continually a threat that initial mission attempt and cost estimation may not be correct. This might bring about multiple changes to estimation in subsequent estimation.
* Lesser design and documentation: Since construct and test starts early, there is always a threat that proper designing and documentation may take a back seat.
* Need clear patron vision: As client input is required in all iterations, it is very important that patron have to have clear vision of quit product. Project can easily lose its route if purchaser have simplest vague idea of the product they want.
* Necessity of skilled and senior assets: Since Agile method is more about less making plans and greater choice making, it is really essential to have experienced and senior resources inside the group or skilled resources mentoring new assets.

# Version Control

Version control systems are a class of computer code tools that facilitate a software team manage changes to ASCII text file over time. Version management computer code keeps track of each modification to the code in an exceedingly special quite info. If a slip is created, developers will flip back the clock and compare earlier versions of the code to assist fix the error whereas minimizing disruption to any or all team members.

Developing computer code while not exploitation version management is risky, like not having backups. Version management may also change developers to maneuver quicker and it permits computer code groups to preserve potency and nimbleness because the team scales to incorporate a lot of developers.



Version control Systems (VCS) have seen nice enhancements over the past few decades and a few are higher than others. VCS are typically called SCM (Source Code Management) tools or RCS (Revision management System). one in every of the foremost in style VCS tools in use nowadays is named bum. bum could be a Distributed VCS, a class called DVCS, a lot of thereon later. Like several of the foremost in style VCS systems out there nowadays, bum is free and open supply. in spite of what they're referred to as, or that system is employed, the first advantages you must expect from version management are as follows.

* A complete long-term change history of every file
* Branching and merging. Having team members work concurrently is a no-brainer, but even individuals working on their own can benefit from the ability to work on independent streams of changes.
* Traceability

## 3.1 Why Use a Version Control System?

right here are many advantages of using a version control device for your projects. This bankruptcy explains some of them in detail.

**Collaboration**

Without a VCS in place, you're probably working together in a shared folder on the same set of documents. Shouting through the office that you are presently running on file "xyz" and that, meanwhile, your teammates ought to preserve their arms off is not a suitable workflow. It's extremely error-prone as you're basically doing open-heart surgery all the time: sooner or later, someone will overwrite a person else's adjustments.

With a VCS, everybody on the crew is able to work certainly freely - on any file at any time. The VCS will later let you merge all of the changes into a commonplace model. There's no question in which the contemporary version of a document or the whole task is. It's in a not unusual, primary place: your model control computer.

**Storing Versions (Properly)**

Saving a model of your project after making adjustments is an vital habit. But without a VCS, this becomes tedious and complicated very quickly:

How a great deal does you save? Only the changed documents or the complete assignment? In the first case, you'll have a tough time viewing the complete task at any point in time - in the latter case, you'll have massive amounts of unnecessary records lying on your hard drive.

How do you name these variations? If you're a completely organized person, you is probably capable of persist with an definitely understandable naming scheme (if you're glad with "acme-inc-redesign-2013-11-12-v23"). However, as soon as it involves variants (say, you want to prepare one model with the header vicinity and one without it), probabilities are good you'll eventually lose track.

The most essential question, however, might be this one: How do you understand what precisely is unique in these variations? Very few people surely make the effort to carefully record each essential exchange and encompass this in a README document within the undertaking folder.

**Restoring Previous Versions**

Being able to restore older versions of a record (or even the whole mission) successfully means one thing: you cannot mess up! If the modifications you've made these days prove to be garbage, you could sincerely undo them in some clicks. Knowing this should make you a lot more relaxed when working on essential bits of a challenge.

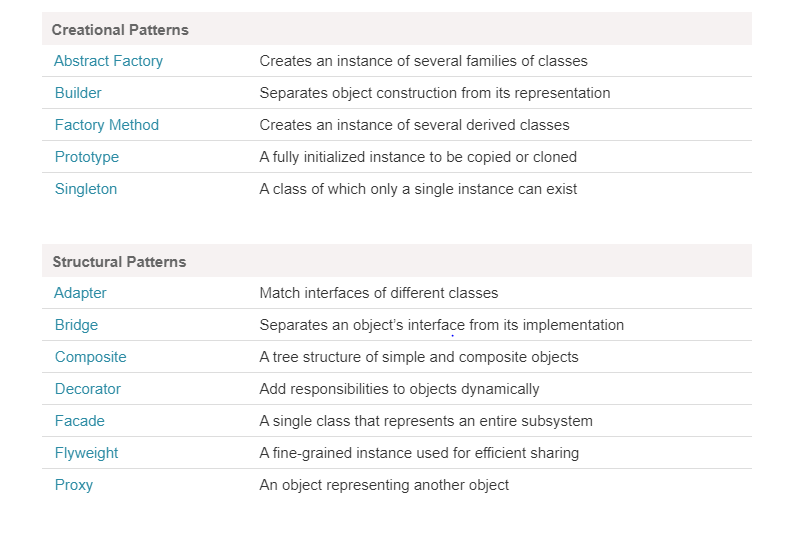
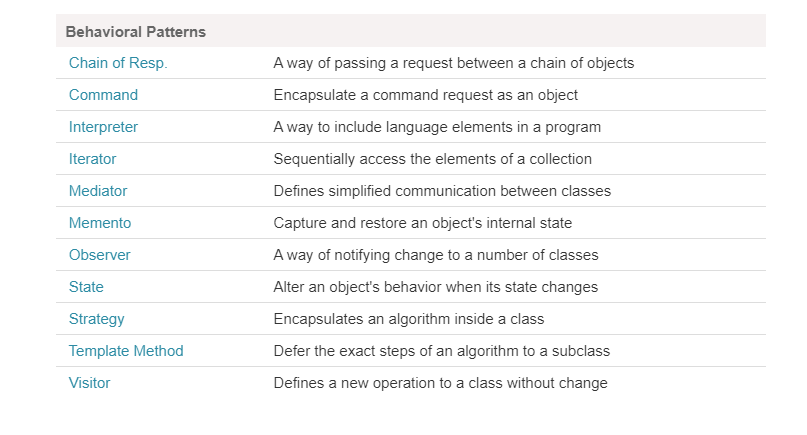
**Backup**

A side-effect of using a allotted VCS like Git is that it can act as a backup; every team member has a full-blown version of the task on his disk - inclusive of the task's whole history. Should the one that you love significant server destroy down (and your backup drives fail), all you need for healing is certainly one of your teammates' local Git repository.

# Design pattern

Christopher Alexander says, "Each sample describes a problem which happens over and once more in our environment, after which describes the center of the solution to that hassle, in such a manner that you can use this solution a million times over, without ever doing it the same way twice" [AIS+77]. Even although Alexander was speak me about patterns in homes and towns, what he says is authentic about object-oriented design patterns. Our solutions are expressed in terms of objects and interfaces as opposed to partitions and doors, but at the center of both kinds of patterns is a approach to a problem in a context. In general, a pattern has four vital elements:

A design pattern names, abstracts, and identifies the key aspects of a common design shape that make it beneficial for developing a reusable object-oriented design. The design pattern identifies the participating lessons and instances, their roles and collaborations, and the distribution of responsibilities. Each design pattern focuses on a particular object-oriented design hassle or issue. It describes whilst it applies, whether it is able to be carried out in view of other design constraints, and the consequences and trade-offs of its use. Since we must ultimately implement our designs, a design pattern also provides pattern C++ and (sometimes) Smalltalk code to demonstrate an implementation. Although design patterns describe object-orientated designs, they are based on realistic solutions which have been carried out in mainstream object-oriented programming languages like Smalltalk and C++ as opposed to procedural languages (Pascal, C, Ada) or extra dynamic object-orientated languages (CLOS, Dylan, Self). We selected Smalltalk and C++ for pragmatic reasons: Our day-to-day experience has been in these languages, and they're increasingly more popular.



## 4.1 Elements of a Pattern

Name

– Important because it becomes part of a design vocabulary

– Raises level of communication

• Problem

– When the pattern is applicable

• Solution

– Design elements and their relationships

– Abstract: must be specialized • Consequences

– Tradeoffs of applying the pattern

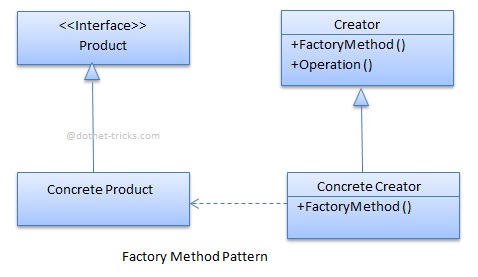
* + Each pattern has costs as well as benefits
  + Issues include flexibility, extensibility, etc.

• There may be variations in the pattern with different consequences

## 4.2 Factory Design Pattern

According to Gang of Four Definition “Define an interface for creating an object, but let the subclasses determine which elegance to instantiate. The Factory method lets a class defer instantiation it uses to subclasses”.

Let simplify the above definition. The Factory Method Design Pattern is used, when we want to create the object (i.E. instance of the Product class) without exposing the object advent common sense to the client. To achieve this, within the factory approach design pattern we can create an abstract class because the Factory magnificence with a view to create and return the instance of the product, but it's going to allow the subclasses determine which magnificence to instantiate. If this isn't clear at the moment then don’t worry, I will explain this with one real-time example.



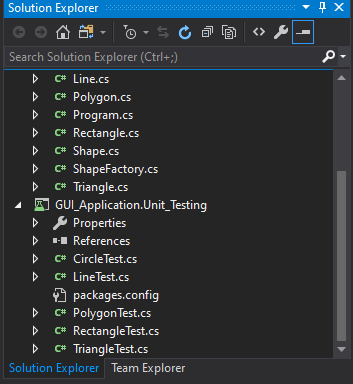
# Unit Testing

Unit testing is a product improvement process in which the littlest testable pieces of an application, called units, are separately and freely examined for legitimate activity. This testing philosophy is finished during the improvement procedure by the product designers and once in a while QA staff. The fundamental target of unit testing is to seclude composed code to test and decide whether it fills in as planned.

Unit testing is a significant advance in the improvement procedure, supposing that done effectively, it can help identify early imperfections in code which might be increasingly hard to discover in later testing stages.

## 5.1 Advantage of unit testing

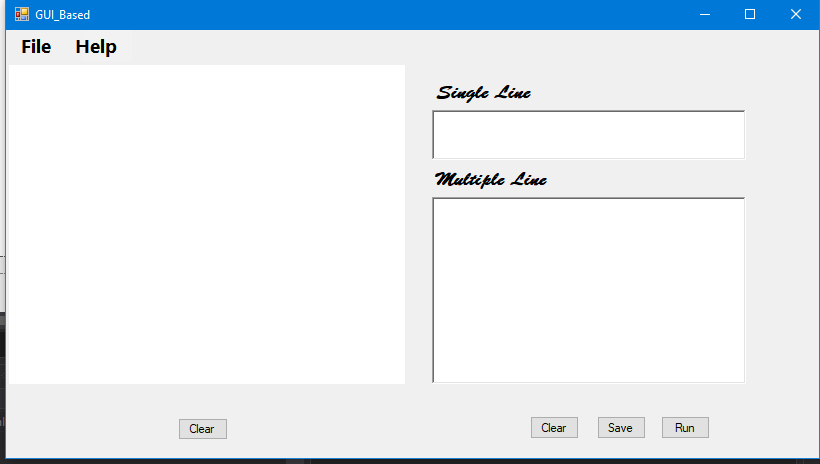
1. Reduces Defects in the newly developed features or reduces bugs when changing the existing functionality.
2. Reduces Cost of Testing as defects are captured in very early phase.
3. Improves design and allows better refactoring of code.
4. Unit Tests, when integrated with build gives the quality of the build as well.



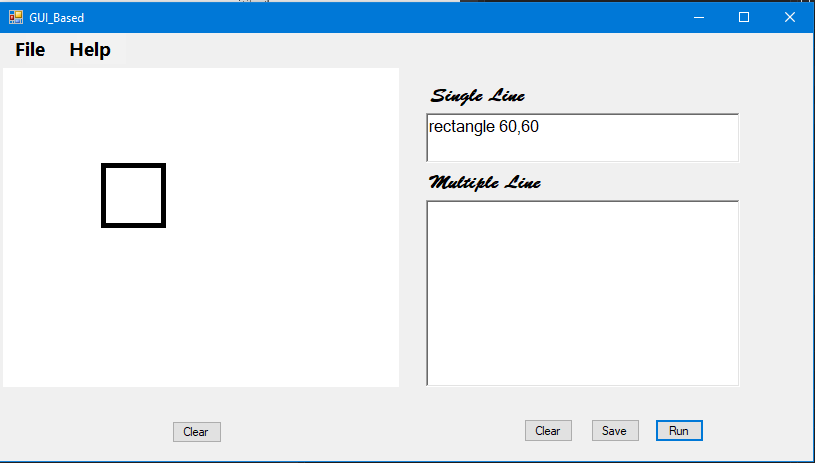
Implementation

Reads and executes commands on command line one at a time

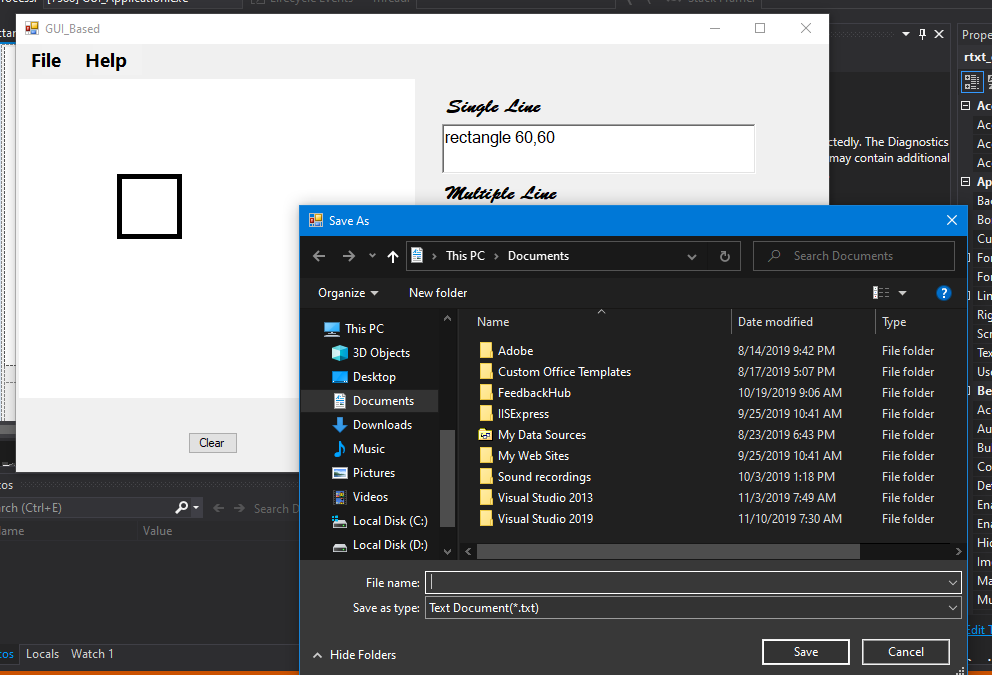
When typing command in textbox it reads and while clicking on run button it displays the result as shown below.



Reads a program and executes it with a “run” command



Saves and loads a program

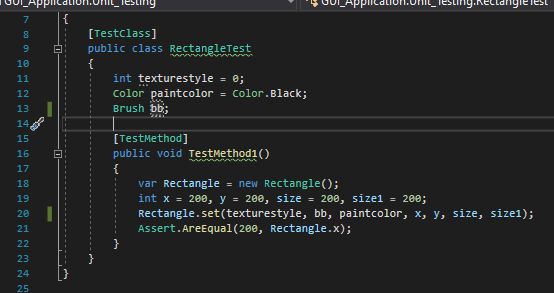


Syntax checking

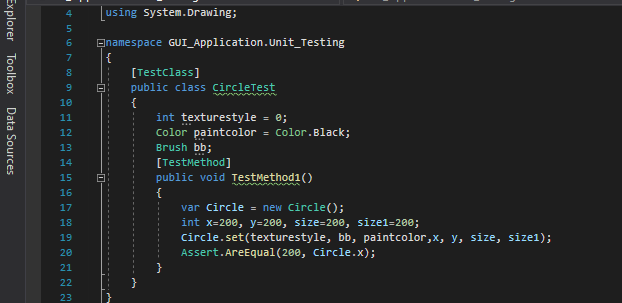
Checks for valid commands

Checks for valid parameters

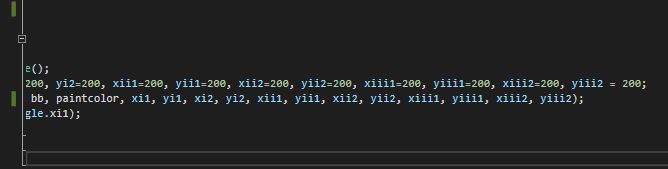
Rectangle (width, height)



Circle(radius)

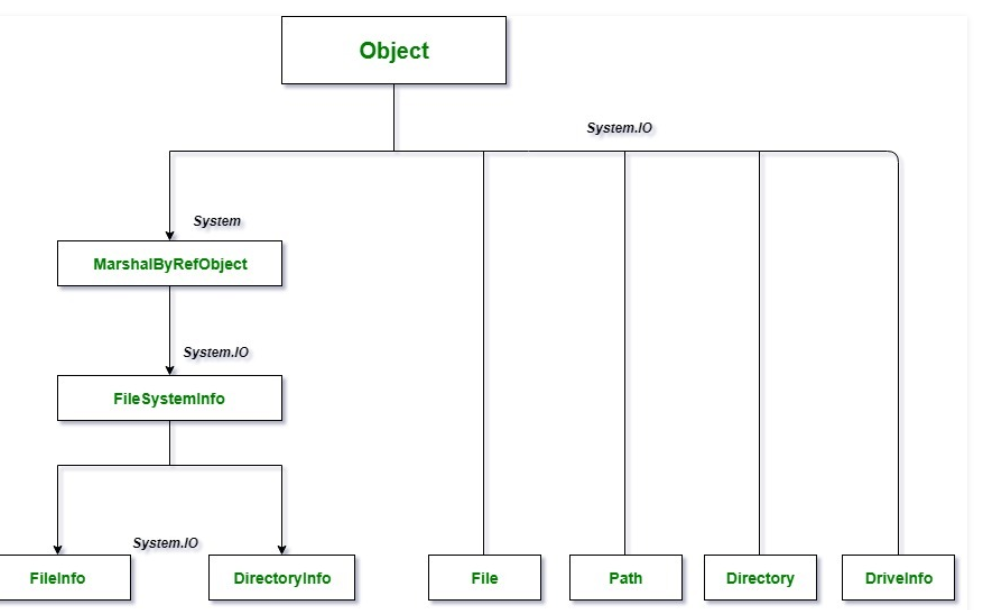


Triangle(side,side,side)



# File Handling in C#

Generally, the report is used to keep the data. The time period File Handling refers to the diverse operations like developing the file, reading from the document, writing to the report, appending the record, etc. There are two fundamental operations that's commonly used in report coping with is reading and writing of the record. The document will become circulate whilst we open the report for writing and studying. A movement is a sequence of bytes that is used for communication. Two circulate can be formed from record one is input stream that's used to study the file and every other is output stream is used to write down in the record. In C#, System.IO namespace includes classes which handle input and output streams and provide information about document and listing structure.



# Handling and raising events

Events in .NET are primarily based on the delegate version. The delegate version follows the observer layout pattern, which permits a subscriber to sign up with and get hold of notifications from a provider. An event sender pushes a notification that an event has happened, and an event receiver receives that notification and defines a response to it. This article describes the major additives of the delegate model, the way to consume activities in applications, and how to enforce activities in your code.

For records about dealing with activities in Windows 8.X Store apps, see Events and routed activities overview.

## 7.1 Events

An event is a message sent by an object to signal the prevalence of an action. The action may be caused by consumer interaction, such as a button click, or it may end result from some other application logic, which includes converting a property’s value. The item that increases the event is referred to as the event sender. The event sender doesn't recognize which item or technique will receive (handle) the activities it increases. The event is usually a member of the event sender; for example, the Click event is a member of the Button magnificence, and the PropertyChanged event is a member of the class that implements the INotifyPropertyChanged interface.

To outline an event, you operate the C# event or the Visual Basic Event keyword within the signature of your event magnificence, and specify the type of delegate for the event. Delegates are described in the next section.

The following example shows how to declare an event named ThresholdReached. The event is associated with the [EventHandler](https://docs.microsoft.com/en-us/dotnet/api/system.eventhandler) delegate and raised in a method named OnThresholdReached.

class Counter

{

public event EventHandler ThresholdReached;

protected virtual void OnThresholdReached(EventArgs e)

{

EventHandler handler = ThresholdReached;

handler?.Invoke(this, e);

}

// provide remaining implementation for the class

}

## 7.2 Delegates

A delegate is a kind that holds a reference to a method. A delegate is said with a signature that suggests the return type and parameters for the strategies it references, and it can maintain references only to techniques that fit its signature. A delegate is thus equivalent to a kind-safe characteristic pointer or a callback. A delegate assertion is sufficient to outline a delegate class.

Delegates have many makes use of in .NET. In the context of events, a delegate is an intermediary (or pointer-like mechanism) among the occasion source and the code that handles the occasion. You partner a delegate with an event by using which include the delegate type within the event assertion, as shown in the instance in the preceding section. For more statistics about delegates, see the Delegate class.

## 7.3 Event handlers

To reply to an event, you outline an event handler technique in the event receiver. This approach have to in shape the signature of the delegate for the event you're handling. In the event handler, you carry out the actions that are required while the occasion is raised, such as amassing user enter after the person clicks a button. To acquire notifications while the occasion occurs, your occasion handler approach must subscribe to the event.

The following example indicates an event handler technique named c\_ThresholdReached that suits the signature for the EventHandler delegate. The method subscribes to the ThresholdReached occasion.

class Program

{

static void Main()

{

var c = new Counter();

c.ThresholdReached += c\_ThresholdReached;

// provide remaining implementation for the class

}

static void c\_ThresholdReached(object sender, EventArgs e)

{

Console.WriteLine("The threshold was reached.");

}

}

# Reference

Anon, (2020). [ebook] Available at: https://www.cs.cmu.edu/~aldrich/courses/15-214-11fa/slides/11-design-patterns.pdf http://sdmeta.gforge.inria.fr/FreeBooks/ByExample/25%20-%20Chapter%2023%20-%20Model-View-Controller.pdf [Accessed 12 Dec. 2019].

Docs.microsoft.com. (2020). *Handling and Raising Events*. [online] Available at: https://docs.microsoft.com/en-us/dotnet/standard/events/ [Accessed 22 Dec. 2019].

Ftms.edu.my. (2020). [online] Available at: http://www.ftms.edu.my/images/Document/IMM006%20-%20RAPID%20APPLICATION%20DEVELOPMENT/Chapter%203%20RAD%20Notes.pdf [Accessed 15 Dec. 2019].

GeeksforGeeks. (2020). *C | File Handling | Question 4 - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/c-file-handling-question-4/ [Accessed 19 Dec. 2019].

Learn.org. (2020). *What Is Software Programming?*. [online] Available at: https://learn.org/articles/What\_is\_Software\_Programming.html [Accessed 16 Dec. 2019].

Tutorialspoint.com. (2019). *Design Pattern - Factory Pattern - Tutorialspoint*. [online] Available at: https://www.tutorialspoint.com/design\_pattern/factory\_pattern.htm [Accessed 16 Dec. 2019].