.NET Core 7.0 Combo Package Includes

1. CSharp 10.0
2. ASP.NET Core 7.0
3. SQL Server 2022
4. Git Hub for developers

C# Programming

Start learning how to program applications using the C# programming language and become a professional from beginner/novice user.

About the Course & Importance of C#

This is an introduction & programming course using Visual C#.NET. It is purely aimed at complete beginners and assumes that you do not have any programming experience whatsoever. No special software is required to buy for the course. You can use the free Visual Studio community edition from Microsoft. You will be guided about which version is required and how to install the same.

The course is all about learning how to develop applications starting from fundamental stage to real world using the C# programming language. Why use C# instead of C++, Java, ActionScript, or some other programming language you may have heard of? First, using C# lets us use the Microsoft Open Source .NET Framework and many 3rd party frameworks, which help us quickly develop applications for mainly Windows and other platforms too. Second, it is the main pillar for building many other applications in .NET for various platforms like building web apps using ASP.NET Web Forms and MVC, Mobile applications using windows phone sdk, Game applications using XNA & XBOX SDK Framework, Cloud applications using Windows Azure and many more And finally, C# is a really good language for learning how to program.

That learning how to program comment is important because our course doesn't assume you have any previous programming experience. Don't worry if you've never written code before; we'll start at the very beginning and work our way up to building industry oriented applications along with project by the end of the course. Throughout the course you'll learn core programming concepts that apply to lots of programming languages, including C#, and you'll also learn how to apply those concepts when you develop different types of applications like Console Applications, classical windows forms applications, modern Windows Presentation Foundation & Distributed programming by combining all of them.

Computer programming is really fun in general, and programming with C# Language using Visual Studio is even better & simpler!

Learning C#:

There are 4 phases to learn C#.

1. Introduction to .NET Core & Language Fundamentals
2. Application Development and C# 10.0 new features
3. Files, Database's, XML & Cloud Storage
4. Distributed Programming

Pre-Requisites	Duration : 45 days
Course Materials & Textbooks : 1. Running Notes in class, 2. Pdfs, 3. HandOuts, 4. Certification Material in Pdfs will be provided for this course.	
Reference Websites:www.dotnetgurukul.com	
Instructor: Praveen Kumar M Learning from Praveen Sir, is always different and you will experience it within few sessions – you attend and work regularly as per given guidance then you will be best in any team that you work in any company – assured.	

C# Course Syllabus by Praveen

Phase I – Introduction to .NET Core & Language Fundamentals

- **.NET Core & Languages**
  - Introduction & Evolution of .NET Core
  - .NET Core, Platform and Technology
  - Pros & Cons of .NET
  - Comparisons .Net Framework Vs .Net Core Vs .Net Standard
  - .NET Languages – C#, VB & others
  - Expressing C# models in UML
  - Building Project with .NET Core
- **C# & its Role in .NET Development**
  - Comparing different versions of C#
  - as Basic Language
  - Unique Features – Current Trend
  - Importance in .NET Development
- **Visual Studio .NET, TFS, GitHub & others**
  - IDE software's & their role
  - Team Development for complete SDLC
  - TFS, GitHub as most essential
  - Installing, configuring lively with local & remote development
  - Understanding complete project environment
- **Programming Basics using C# & VB.NET**
  - Console Applications
  - Current & Latest Versions
  - Program structure
  - Solution – Project – files
  - Type System in .NET – CTS & CLS , NET Core CLI , .NET Standards
  - CSC,VBC compilers etc
  - Programming Structures
- **Data types and control constructs**
  - Declaring Implicit and Explicit variables
  - Value and Reference types
  - Unicode characters and strings
- **Defining and calling methods**
  - The Main method specification
  - Passing arguments and returning values
  - The scope and lifetime of variables
  - Named and symbolic methods
  - Handling exceptions
  - Recovering resources
- **Object Oriented Programming – I**
  - Need for Object Oriented & current programming Model
  - .NET as Component Oriented Development
  - Creating Classes & Objects according to Component Specification
  - Developing C# Classes
  - Defining classes
  - Encapsulating attributes with methods and properties
  - Providing consistent initialization using constructors
  - Overloading methods and constructors
  - Achieving reuse through inheritance and polymorphism

Creating and using objects

- Allocating object memory with new
- Passing initial values to constructors

- Choosing value or reference allocation
- Invoking methods and accessing properties

Object Oriented Programming – II

Exposing interfaces

- Defining an interface specification
- Interface polymorphism
- Events and delegates
- Classes and Methods, Classes and Inheritance
- Namespaces and Referencing Libraries
- Understanding Scope and Utilizing Access Modifiers

Component Development of .NET

- Component Features of .NET
- Manifests and Assemblies
- .NET Assembly and MetaModel
- Creating and calling custom components
- Extending System.ComponentModel.Component
- Interfacing legacy components
- Accessing COM/DCOM
- Tools for forward and backward compatibility
- Calling existing components
- Coordinating components through the CLI
- Accessing metadata
- Handling cross-language differences

Handling Exceptions

- Error Handling and its importance
- Exceptions Hierarchy
- Creating and throwing exception types
- Multiple Exception Types.

Phase II – Application Development and C# 10.0 new features

Working with Collections & Generics

- ArrayList class
- Generics and behaviour
- List class, Generics and List
- Dictionary class
- HashTable,Stack and other collection classes

Starting with classical Windows Forms Development

- Understanding GUI Development
- Understanding Event Driven Programming
- Working with controls – Types of Windows Forms Controls

Working with WPF as current Desktop development

- WPF as more preferred Desktop Development
- XAML as new Design Markup Language
- WPF Controls & Concepts

Combining Object Oriented Concepts with WPF

- Building a simple module

SOLID Principles in C#

- Understand the basic concepts and principles of software architecture
- Learn how to organize code in different layers and tie them together
- Recognize the 4 Meta-principles of software architecture
- Demonstrate an understanding of the SOLID principles of object-oriented design
- Develop simple yet powerful C# code that is based on the above principles
- Outline the best practices of software architecture design

- Construct maintainable and extendable code that is easy to debug, modify and reuse
- Find new ways to apply the concepts of software architecture to create efficient applications

C# 10.0 New Features

- Record structs
- Improvements of structure types
- Interpolated string handlers
- global using directives
- File-scoped namespace declaration
- Extended property patterns
- Improvements on lambda expressions
- Allow const interpolated strings
- Record types can seal ToString()
- Improved definite assignment
- Allow both assignment and declaration in the same deconstruction
- Allow AsyncMethodBuilder attribute on methods
- CallerArgumentExpression attribute
- Enhanced #line pragma
- Warning wave 6

Phase III (Files , Database' s, XML & Cloud Storage)

Working with Storage – Understand various storage options

Files & Streams as most basic & important storage options

- File Handling in C#.
- Streams & some basic Classes.
- System.IO

ADO.NET to work with Database's

- Databases overview from C#.
- ADO.Net and C#.
- ADO.Net as the new model for building db applications. (old DAO,RDO)
- Advantages and Disadvantages of ADO.Net.
- Different ADO.NET versions starting from v1.0 to 7.0
- Drawbacks of ADO.NET 2.0 model and new models available for development starting from v3.0 to v4.0.
- Starting with ADO.NET 2.0
- System.Data, System.Data.oledb, System.data.sqlclient, System.Data.OracleCl namespaces…..
- Command Object in ADO.Net, Running commands and Stored Procedures using Command Object.
- Datasets, In and Out of Datasets. Complete usage with examples.
- Layers & N-Tier architecture. Demo combining OOPS, ADO.NET & UI.
- Database Concepts Continued…

ADO.NET Linq To Sql

- ORM implementation for 1st time in MS Environment
- ORM detailed – its advantages and disadvantages
- LINQ and its importance in .NET
- Extension methods
- Object & collection initialization syntax

- Anonymous types
- Lambda expressions
- Various Scenarios with Linq to Sql

- **ADO.NET Entity Framework**
  - Rich Implementation of ORM
  - Most advanced environment for DB handling
  - Design Patterns support and implementation
  - Repository, Unit of Work patterns
  - Linq , Linq to Entities in Detail
- **XML, System.Xml and its classes**
  - XML and its uses.
  - DataSet with XML
  - System.XMl namespace and its consumption in our applications.
  - XmlTextReader, XmlTextWriter, XmlDocument as key objects.
- **LINQ To XML**
- **Understanding Cloud Storage & programming with Cloud Storage**

Phase IV (Distributed Programming)

- **Building Multi-Tier applications**
  - Leveraging solid architectures (MVC and EDM)
  - Substituting the user interface
  - Coding industry-standard design patterns in C#
- **Windows Services**
  - Understanding Windows Services
  - Creating Windows Services
  - Installing and Uninstalling Windows services
- **DCOM, RMI & CORBA**
- **.NET Remoting ---Micros servies**
  - Understanding Distributed Architecture
  - Drawbacks of DCOM
  - Remoting Advantages
  - Remoting Vs web services
  - Creating & Using Remote Applications
- **WCF as preferred Distributed Environment**
- **Remoting/WCF in .NET**
- **Deployment of Applications**
  - **Setup and Packaging of Applications**
  - **Fast Deployment Methods**

ASP.NET Core Programming

Start learning how to program ASP.NET MVC Core applications using the C# programming language and become a professionalfrom beginner/novice user.

About the Course & Importance of ASP.NET Core

All .NET Beginner(s)/Professional(s) who are keen to develop modern, light weight and cloud based web applications should go for this course. ASP.NET Web form and ASP.NET MVC available because of its age is considered to be very matured for web application development, it is because of the popularity of ASP.NET Core Framework and many exclusive features of ASP.NET Core that today it is dominating over Web Forms and MVC are first choice for large sized enterprise web application development.

Computer programming is really fun in general, and programming with C# Language using Visual Studio is even better & simpler!

Learning ASP.NET Core :

There are 3 phases to learn ASP.NET Core.

1. Introduction to ASP.NET Core Fundamentals
2. ASP.NET Core Middleware
3. ASP.NET Core 7.0 Advanced Topics

Pre-Requisites: C# basic Knowledge Duration : 45 days
Course Materials & Textbooks : 1. Running Notes in class, 2. Pdfs, 3. HandOuts, 4. Certification Material in Pdfs will be provided for this course.
Reference Websites:www.dotnetgurukul.com
Instructor: Praveen Kumar M Learning from Praveen Sir, is always different and you will experience it within few sessions – you attend and work regularly as per given guidance then you will be best in any team that you work in any company – assured.

ASP.NET Core Course Syllabus by Praveen

Phase I – Introduction to ASP.NET Core

- **ASP.NET Core Introduction**
  - Introduction & Evolution of ASP.NET Core (Brief history of ASP , ASP.NET , ASP.NET MVC)
  - What is ASP.NET Core
  - ASP.NET Core Features
  - Advantages of ASP.NET Core
  - MVC Pattern
  - Understanding ASP.NET Core MVC
  - ASP.NET Core vs. ASP.NET MVC vs. ASP.NET Web Forms

  (Introducing Markup langs .html , .xhtml ,.xml , aspx , .cshtml)
- **ASP.NET Core First Application**
  - ASP.NET Core Environment Setup
  - ASP .NET Core First Application
  - Project Layout
  - Understanding Life Cycle of ASP.Net Core Request
- **Controllers**
  - Controllers and Action Methods Overview
  - Action Methods and IActionResult object
  - Passing data from Controller to View
  - Understanding Action Selectors
  - Action Filters
  - Building Custom Action Filters
  - Middleware
  - Asynchronous Action Methods
- **Views**
  - Introducing Razor View
  - Advantages of Razor View
  - Razor Syntax
  - Types of Views
  - Partial Views
  - Layout Pages
  - Special Views
  - View Categorization based on Model
- **Model Binding**
  - Html Form behavior
  - Model Binder Overview
  - DefaultModelBinder
  - Binding to Complex Classes
  - IFormCollection Model Binding
  - IFormFile Model Binder
  - Bind Attribute
  - TryUpdateModelAsync

Phase 2 – ASP.NET Core Middleware

- **Helpers**
  - Html Helpers
  - Built-In Html Helpers
  - URL helpers
  - Tag Helpers
  - Custom Tag Helpers
- **Validations & Data Annotations**
  - Data Annotations and Validations Overview
  - Validations with Data Annotation
  - Server Side and Client-Side Validation
  - Custom Server-side validation
  - Model level validation using IValidatableObject
  - Custom unobstrive Client-side Validation
  - Remote Validation
- **State management Techniques**
  - Cookies
  - Sessions
- **Routing**
  - Url Routing Overview
  - Custom Routes
  - Attribute Routing
  - Routing Constraints
- **Security**
  - Authentication and Authorization
  - Implementing Security using ASP.NET Core Identity
- **MVC and Entity Framework Core**
  - Basic CRUD Operations using Entity Framework
  - Separation of work using BO Classes
  - Writing Generic Class / Repository
  - Caching in Repository
- **ASP.NET Core - Web Caching**
  - Cache Tag Helpers
  - Memory Caching Introduction
  - In-Memory Caching
  - Response Cache
  - Distributed Cache
- **Module Development**
  - Understanding Areas
  - Adding Areas
  - Defining Area Routes
  - Linking between Areas
- **Web API and JQuery Ajax**
  - Introduction to Web API
  - AJAX implementation using JQuery
  - Calling the Web API with JQuery Ajax
  - Creating a Web API that Supports CRUD Operations using EF
- **Bundling & Minification**
  - What is Bundling and Minification in ASP.net Core?
  - Bundler and Minifier Extension
  - How to Bundle your files
  - How to minify your Bundles
  - Convert to Gulp

Phase 3 – Upgrading concepts of ASP.NET Core 7.0

ASP.NET Core 7.0 Changes

- Merger of the Program and Startup classes
- Program.cs file changes

ASP.NET Core MVC and Razor improvements

- Minimal APIs . Create a minimal web API with ASP.NET Core
- Minimal APIs quick reference
- Differences between minimal APIs and APIs with controllers
- Code samples migrated to the new minimal hosting model in 7.0

ASP.NET Core performance and API improvements

- Non-allocating app.Use extension method.
- Reduced memory allocations when accessing HttpRequest.Cookies.
- Use LoggerMessage.Define for the windows only HTTP.sys web server
- Reduce the per connection overhead in SocketConnection by ~30%.
- Reduce allocations by removing logging delegates in generic types.

HTTP Logging middleware

- Support for the HTTP Logging middleware has been introduced in ASP.NET Core 6.
- You can take advantage of this middleware in ASP.NET Core 6 to log information about HTTP requests and responses that include one or more of the following.
- Request information
- Response information
- Request and response headers
- Body of the request
- Properties

SQL SERVER 2022 Database

Start learning how to work with SQL Server 2022

About the Course & Importance of SQL SERVER 2022

This SQL Server training teaches developers all the Transact-SQL skills they need to create database objects like Tables, Views, Stored procedures & Functions and triggers in SQL Server. Gives idea about writing Queries & Sub-queries, working with Joins, etc. As well as database management skills like backup, restore, etc.

There are 4 phases to learn SQL SERVER 2022.

1. Introduction to Databases
2. Querying Data with Transact-SQL
3. Developing SQL Databases
4. Updating Your Skills to SQL Server 2022

Computer programming is really fun in general.

Pre-Requisites : No Prior Experience is Presumed. Duration : 45 days
Course Materials & Textbooks : 1. Running Notes in class, 2. Pdfs, 3. HandOuts, 4. Certification Material in Pdfs will be provided for this course.
Reference Websites:www.dotnetgurukul.com
Instructor: **Praveen Kumar M** Learning from Praveen Sir, is always different and you will experience it within few sessions – you attend and work regularly as per given guidance then you will be best in any team that you work in any company – assured.

SQL SERVER Course Syllabus by Praveen Kumar M

Phase I - Introduction to Databases

- **Introduction to Databases**
  - Introduction to Relational Databases
  - Other Types of Databases and Storage
  - Data Analysis
  - Database Languages in SQL Server
  - Lab: Exploring and Querying SQL Server Databases using T-SQL

- **Introduction to Microsoft SQL Server 2022 Tools**
  - The Basic Architecture of SQL Server 2022
  - SQL Server Editions and Versions
  - Getting Started with SQL Server Management Studio
  - Lab: Working with SQL Server 2022 Tools

- **Introduction to SQL Server 2022 Installation**
  - SQL Server 2022 Editions and Components
  - Installing SQL Server 2022
  - SQL Server Management Studio Enhancements
  - Lab: Exploring SQL Server 2022

- **Data Modeling**
  - Data Modeling
  - ANSI-SPARC Database Model
  - Entity Relationship Modeling
  - Lab: Identify Components in Entity Relationship Modeling

- **Normalization**
  - Fundamentals of Normalization
  - Normal Form
  - Denormalization
  - Lab: Normalizing Data

- **Relationships and Types**
  - Introduction to Relationships
  - Planning Referential Integrity
  - Lab: Planning and Implementing Referential Integrity

- **Performance**
  - Indexing
  - Query Performance
  - Concurrency
  - Lab: Performance Issues

- **Introduction to Database Objects**
  - Tables
  - Views
  - Stored Procedures, Triggers, and Functions
  - Lab: Using SQL Server

Phase II – Querying Data with Transact-SQL

- **Introduction to T-SQL Querying**

- Introducing T-SQL
- Understanding Sets
- Understanding Predicate Logic
- Understanding the Logical Order of Operations in SELECT Statements
- Lab: Introduction to T-SQL Querying

Writing SELECT Queries

- Writing Simple SELECT Statements
- Eliminating Duplicates with DISTINCT
- Using Column and Table Aliases
- Writing Simple CASE Expressions
- Lab: Writing Basic SELECT Statements

Querying Multiple Tables

- Understanding Joins
- Querying with Inner Joins
- Querying with Outer Joins
- Querying with Cross Joins and Self Joins
- Lab: Querying Multiple Tables

Sorting and Filtering Data

- Sorting Data
- Filtering Data with Predicates
- Filtering Data with TOP and OFFSET-FETCH
- Working with Unknown Values
- Lab: Sorting and Filtering Data

Working with SQL Server Data Types

- Introducing SQL Server Data Types
- Working with Character Data
- Working with Date and Time Data
- Lab: Working with SQL Server 2016 Data Types

Using DML to Modify Data

- Adding Data to Tables
- Modifying and Removing Data
- Generating Automatic Column Values
- Lab: Using DML to Modify Data

Using Built-In Functions

- Writing Queries with Built-In Functions
- Using Conversion Functions
- Using Logical Functions
- Using Functions to Work with NULL
- Lab: Using Built-in Functions

Grouping and Aggregating Data

- Using Aggregate Functions
- Using the GROUP BY Clause
- Filtering Groups with HAVING
- Lab: Grouping and Aggregating Data

Using Subqueries

- Writing Self-Contained Subqueries
- Writing Correlated Subqueries
- Using the EXISTS Predicate with Subqueries
- Lab: Using Subqueries

Using Table Expressions

- Using Views
- Using Inline TVFs
- Using Derived Tables
- Using CTEs
- Lab: Using Table Expressions

Using Table Expressions

- Using Views
- Using Inline TVFs
- Using Derived Tables
- Using CTEs
- Lab: Using Table Expressions

Using Set Operators

- Writing Queries with the UNION Operator
- Using EXCEPT and INTERSECT
- Using APPLY
- Lab: Using Set Operators

Using Window Ranking, Offset, and Aggregate Functions

- Creating Windows with OVER
- Exploring Window Functions
- Lab: Using Window Ranking, Offset, and Aggregate Functions

Pivoting and Grouping Sets

- Writing Queries with PIVOT and UNPIVOT
- Working with Grouping Sets
- Lab: Pivoting and Grouping Sets

Executing Stored Procedures

- Querying Data with Stored Procedures
- Passing Parameters to Stored Procedures
- Creating Simple Stored Procedures
- Working with Dynamic SQL
- Lab: Executing Stored Procedures

Implementing Error Handling

- Implementing T-SQL Error Handling
- Implementing Structured Exception Handling
- Lab: Implementing Error Handling

- **Implementing Transactions**
  - Transactions and the Database Engine
  - Controlling Transactions
  - Lab: Implementing Transactions

Phase III (Developing SQL Databases)

- **Designing and Implementing Tables**
  - Designing Tables
  - Data Types
  - Working with Schemas
  - Creating and Altering Tables
  - Lab: Designing and Implementing Tables

- **Advanced Table Designs**
  - Partitioning Data
  - Compressing Data
  - Temporal Tables
  - Lab: Using Advanced Table Designs
- **Introduction to Indexes**
  - Core Indexing Concepts
  - Data Types and Indexes
  - Heaps, Clustered, and Non-clustered Indexes
  - Single Column and Composite Indexes
  - Lab: Implementing Indexes

- **Designing Optimized Index Strategies**
  - Index Strategies
  - Managing Indexes
  - Execution Plans.
  - The Database Engine Tuning Advisor
  - Query Store
  - Lab: Optimizing Indexes

- **Columnstore Indexes**
  - Introduction to Columnstore Indexes
  - Creating Columnstore Indexes
  - Working with Columnstore Indexes
  - Lab: Using Columnstore Indexes

- **Designing and Implementing Views**
  - Introduction to Views
  - Creating and Managing Views
  - Performance Considerations for Views
  - Lab: Designing and Implementing Views

- **Designing and Implementing Stored Procedures**
  - Introduction to Stored Procedures
  - Working with Stored Procedures

- Implementing Parameterized Stored Procedures
- Controlling Execution Context
- Lab: Designing and Implementing Stored Procedures

Designing and Implementing User-Defined Functions

- Overview of Functions
- Designing and Implementing Scalar Functions
- Designing and Implementing Table-Valued Functions
- Considerations for Implementing Functions
- Alternatives to Functions
- Lab: Designing and Implementing User-Defined Functions

Responding to Data Manipulation Via Triggers

- Designing DML Triggers
- Implementing DML Triggers
- Advanced Trigger Concepts
- Lab: Responding to Data Manipulation by Using Triggers

Using In-Memory Tables

- Memory-Optimized Tables
- Natively Compiled Stored Procedures
- Lab: Using In-Memory Database Capabilities

Implementing Managed Code in SQL Server

- Introduction to CLR Integration in SQL Server
- Implementing and Publishing CLR Assemblies
- Lab: Implementing Managed Code in SQL Server

Storing and Querying XML Data in SQL Server

- Introduction to XML and XML Schemas
- Storing XML Data and Schemas in SQL Server
- Implementing the XML Data Type
- Using the Transact-SQL FOR XML Statement
- Getting Started with XQuery
- Shredding XML
- Lab: Storing and Querying XML Data in SQL Server

Storing and Querying Spatial Data in SQL Server

- Introduction to Spatial Data
- Working with SQL Server Spatial Data Types
- Using Spatial Data in Applications
- Lab: Working with SQL Server Spatial Data

Storing and Querying BLOBs and Text Documents in SQL Server

- Considerations for BLOB Data
- Working with FILESTREAM
- Using Full-Text Search
- Lab: Storing and Querying BLOBs and Text Documents in SQL Server

Storing and Querying BLOBs and Text Documents in SQL Server

- Considerations for BLOB Data

- Working with FILESTREAM
- Using Full-Text Search
- Lab: Storing and Querying BLOBs and Text Documents in SQL Server

SQL Server Concurrency

- Concurrency and Transactions
- Locking Internals
- Lab: Concurrency and Transactions

Performance and Monitoring

- Extended Events
- Working with Extended Events
- Live Query Statistics
- Optimize Database File Configuration
- Metrics
- Lab: Monitoring, Tracing, and Baselining

Phase IV (Updating Your Skills to SQL Server 2022)

What's New in SQL Server Performance?

- Operational Analytics
- In-Memory OLTP Enhancements
- Query Store
- Live Query Statistics
- Native JSON
- Temporal Tables
- Lab : Implementing SQL Server 2016 Performance Improvements

What's New in SQL Server Security

- Using Always Encrypted
- Row-Level Security
- Dynamic Data Masking
- Lab : SQL Server 2016 Security Improvements

What's New in SQL Server Availability and Scalability?

- Enhanced Always On Availability Groups
- What's New with tempdb?
- Use Windows Server 2022with SQL Server 2022
- Lab : Monitoring tempdb

What's New in SQL Server Reporting and BI

- Reporting Services Enhancements
- Power BI Enhancements
- Mobile Report Publisher
- Lab : Implementing Power BI

What's New in SQL Server Data Access?

- PolyBase
- What's New in Integration Services?
- Working with SSIS and Azure
- Lab : Exploring the New Features of SQL Server Integrated Services (SSIS)

New and Enhanced Features in SQL Server OLAP

- New and Enhanced Features in SQL Server OLAP
- What's New in SQL Server Analysis Services?
- Lab : OLAP with SQL Server

What's New for SQL Server in the Cloud?

- Stretch Database
- Enhanced Backup to Azure
- What's New in Azure SQL Database?
- Lab : Using Stretch Database

GitHub for Developers

Getting Started With Collaboration

What is GitHub?
The GitHub Ecosystem What is Git?
Exploring a GitHub Repository Using GitHub Issues
Activity: Creating A GitHub Issue Using Markdown

Understanding the GitHub

The Essential GitHub Workflow

Branching with Git

Branching Defined
Activity: Creating A Branch with GitHub

Local Git Configuration

Checking Your

Git Version Git

Configuration Levels

Viewing Your

Configurations

Configuring Your User Name

and Email Configuring autocrlf

Working Locally with Git

Creating a Local Copy of the repo

Our Favorite Git command: git

status Using Branches locally

Switching Branches

Activity: Creating a New File The Two Stage Commit

Collaborating on Your Code

Pushing Your Changes to GitHub

Activity: Creating a Pull Request

Exploring a Pull Request Activity:

Code Review

Editing Files on GitHub

Editing a File on GitHub

Committing Changes on GitHub

Merging Pull Requests

Merge Explained

Merging Your Pull Request

Updating Your Local Repository

Cleaning Up the Unneeded Branches

Viewing Local Project History

Using Git Log

Streamlining Your Workflow with Aliases

Creating Custom Aliases

Workflow Review Project: GitHub Games

User Accounts vs. Organization Accounts

Introduction to GitHub Pages

What is a Fork?

Creating a Fork

Workflow Review: Updating the README.md

Resolving Merge Conflicts

Local Merge Conflicts

Working with Multiple Remotes

Remote Merge Conflicts

Exploring

Searching for Events in Your Code

What is git bisect?

Finding the Bug in Our Project

Reverting Commits

How Commits Are Made

Safe Operations

Reverting Commits

Helpful Git Commands

Moving and Renaming Files with Git

Staging Hunks of Changes

Viewing Local Changes

Comparing Changes within the Repository

Creating a New Local Repository

Initializing a New Local Repository

Fixing Commit Mistakes

Revising Your Last Commit

Rewriting History with Git Reset

Understanding Reset

Reset Modes

Reset Soft

Reset Mixed

Reset Hard

Does Gone Really Mean Gone?

Getting it Back

You Just Want That One CommitOops, I

Didn’t Mean to Reset

Merge Strategies: Rebase

About Git rebase

Understanding Git Merge StrategiesCreating

a Linear History