# W06 Assignment: Explain Polymorphism

Polymorphism is a key principle in object-oriented programming that allows a single interface to represent multiple underlying types.   
This means a base class can define a method signature, while derived classes provide their own unique implementations.   
At runtime, the correct method is dispatched based on the actual object type.  
  
Benefits of Polymorphism  
- Flexibility: Enables general code to work with base-class types while allowing specialized behavior in subclasses.  
- Maintainability: Reduces code duplication and simplifies future enhancements.  
- Extensibility: New types can be added without modifying existing logic.  
  
Application in My W06 Project  
In my project, I used a Goal base class with derived classes like SimpleGoal, EternalGoal, and ChecklistGoal.   
Each class overrides the RecordEvent() method to define its own behavior.   
This allows the GoalManager to loop through a list of Goal objects and call RecordEvent() without needing to know the specific type.  
  
Code Example:  
public abstract class Goal  
{  
 public abstract int RecordEvent(); // different goals return different points  
}  
  
public class SimpleGoal : Goal  
{  
 public override int RecordEvent()  
 {  
 \_isComplete = true;  
 return \_points;  
 }  
}  
  
public class EternalGoal : Goal  
{  
 public override int RecordEvent()  
 {  
 return \_points; // repeatable reward, never completes  
 }  
}  
  
Runtime Behavior  
When GoalManager executes:  
int pts = goal.RecordEvent();  
The correct subclass method is called automatically.   
This design decouples the manager from individual goal logic, making the program easier to maintain and scale.