Motivation:

Extract Subclass refactoring needed when a class has features that are used in certain cases. These features are needed to be pushed down to its own class. In this example, class JobItem has methods and fields that are rare use in this class. Because of this rare use it would be efficient to move these methods and fields to a subclass by using this refactoring technique

**Before:**

class JobItem

{

public int Quantity {get; private set;}

public Employee Employee {get; private set;}

public bool IsLabour {get; set;}

private int \_unitPrice;

public int UnitPrice

{

get

{

return IsLabour? Employee.Rate: UnitPrice;

}

set

{

\_unitPrice = value;

}

}

public int TotalPrice

{

get {return UnitPrice \* Quantity;}

}

public JobItem (int unitPrice, int quantity, bool isLabour, Employee employee)

{

UnitPrice = unitPrice;

Quantity = quantity;

IsLabour = isLabour;

Employee = employee;

}

}

class Employee

{

public int Rate {get; internal set;}

}

**Mechanics:**

Here I am going to create class LaborItem which is a subclass of JobItem. Moving designated field from class JobItem to class LaborItem. Make the constructor protected. Moving these fields by **Push Down Method** and **Push Down Field**

**After:**

class JobItem

{

public int Quantity {get; private set;}

public bool IsLabour {get; set;}

public virtual int UnitPrice {get; set;}

public int TotalPrice

{

get {return UnitPrice \* Quantity;}

}

/// <summary>

/// Constructor

/// </summary>

/// <param name="unitPrice"></param>

/// <param name="quantity"></param>

protected JobItem (int unitPrice, int quantity)

{

UnitPrice = unitPrice;

Quantity = quantity;

}

}

class LaborItem: JobItem

{

public Employee Employee {get; protected set;}

/// <summary>

/// Constructor

/// </summary>

/// <param name="quantity"></param>

/// <param name="employee"></param>

public LaborItem (int quantity, Employee employee): base(0, quantity)

{

Employee = employee;

}

public override int UnitPrice

{

get

{

return Employee.Rate;

}

set

{

base.UnitPrice = value;

}

}

}

class Employee

{

public int Rate {get; internal set;}

}