**Step Builder Pattern:-**

* extension of builder pattern..
* isse hum order of function execution define kr skte h kis order function call honge

**Jaise:-**

Suppose shoes phne h toh steps honge:-

* wash ur legs
* wear socks
* wear shoes

isi order m toh same can be achived by step builder pattern

**Code:-**

//  
// Source code recreated from a .class file by IntelliJ IDEA  
// (powered by Fernflower decompiler)  
//  
  
package com.mmt.checkout.pojo.sections;  
  
public interface PaymentSection {  
 String PAYMENT\_TYPE\_PARTIAL = "partialPayment";  
 String MINIMUM\_PARTIAL\_AMOUNT = "minimumPartialAmount";  
 String UATP\_AIRLINE\_CODE = "uatpAirLineCode";  
 String UATP\_PG\_NAME = "uatpPGName";  
 String UATP\_AMOUNT = "uatpAmount";  
  
 PaymentSection.PaymentParam1 setChargingCurrency(String var1);  
  
 public interface GenericPaymentParam {  
 PaymentSection.GenericPaymentParam setAdditionalParameter(String var1, String var2);  
 }  
  
 public interface PaymentParam2 {  
 PaymentSection.GenericPaymentParam setFailureReturnURL(String var1);  
 }  
  
 public interface PaymentParam1 {  
 PaymentSection.PaymentParam2 setSuccessReturnURL(String var1);  
 }  
}

//  
// Source code recreated from a .class file by IntelliJ IDEA  
// (powered by Fernflower decompiler)  
//  
  
package com.mmt.checkout.pojo.sections;  
  
import com.google.gson.GsonBuilder;  
import java.util.Map;  
  
public interface EnhancedPaymentSection extends PaymentSection {  
 void registerTypeAdapters(GsonBuilder var1);  
  
 Map<String, String> getPaymentParameterMap();  
}

//  
// Source code recreated from a .class file by IntelliJ IDEA  
// (powered by Fernflower decompiler)  
//  
  
package com.mmt.checkout.pojo.sections.impl;  
  
import com.google.gson.GsonBuilder;  
import com.mmt.checkout.pojo.sections.EnhancedPaymentSection;  
import com.mmt.checkout.pojo.sections.PaymentSection;  
import com.mmt.checkout.pojo.sections.PaymentSection.GenericPaymentParam;  
import com.mmt.checkout.pojo.sections.PaymentSection.PaymentParam1;  
import com.mmt.checkout.pojo.sections.PaymentSection.PaymentParam2;  
import java.util.LinkedHashMap;  
import java.util.Map;  
  
public class PaymentSectionImpl implements PaymentSection, PaymentParam1, PaymentParam2, GenericPaymentParam, EnhancedPaymentSection {  
 private Map<String, String> paymentParameterMap = new LinkedHashMap();  
  
 public PaymentSectionImpl(String channel, String product, String searchKey) {  
 this.paymentParameterMap.put("channel", channel);  
 this.paymentParameterMap.put("product", product);  
 this.paymentParameterMap.put("searchKey", searchKey);  
 }  
  
 public GenericPaymentParam setAdditionalParameter(String key, String value) {  
 this.paymentParameterMap.put(key, value);  
 return this;  
 }  
  
 public GenericPaymentParam setFailureReturnURL(String failureReturnUrl) {  
 this.paymentParameterMap.put("failureReturnUrl", failureReturnUrl);  
 return this;  
 }  
  
 public PaymentParam2 setSuccessReturnURL(String successReturnUrl) {  
 this.paymentParameterMap.put("successReturnUrl", successReturnUrl);  
 return this;  
 }  
  
 public PaymentParam1 setChargingCurrency(String currency) {  
 this.paymentParameterMap.put("chargingCurrency", currency);  
 return this;  
 }  
  
 public void registerTypeAdapters(GsonBuilder gsonBuilder) {  
 }  
  
 public Map<String, String> getPaymentParameterMap() {  
 return this.paymentParameterMap;  
 }  
}

**Link:-** <http://rdafbn.blogspot.com/2012/07/step-builder-pattern_28.html>

**Cons:-**

**A possibility when some parameters are mandatory is to put them in the constructor of the builder so they are always included while the optional parameters benefit of the builder pattern. This is simpler than a step by step builder.**