LabVIEW

Dr Marko Dimitrijević

Objektno orijentisano programiranje

Objektno orijentisano programiranje

- Objektno orijentisano programiranje
- Prednosti OOP
- OOP razvoj virtuelnih instrumenata u LabVIEW

Objekno orijentisano programiranje

- Objektno orijentisano programiranje je pristup razvoju aplikacija
- OOP je pogodan za razvoj aplikacija i projekata koje realizuje veliki broj programera

Prednosti OOP

Prednosti objektno orijentisnog programiranja (OOP)

- OOP omogućuje ponovnu upotrebu napisanog kôda (code reuse)
- Umanjuje potrebu za izmenom kôda i kompleksnost
- Pojednostavljuje proširenje funkcionalnosti aplikacija

Modularno programiranje u LabVIEW

Modularno programiranje u LabVIEW je podržano:

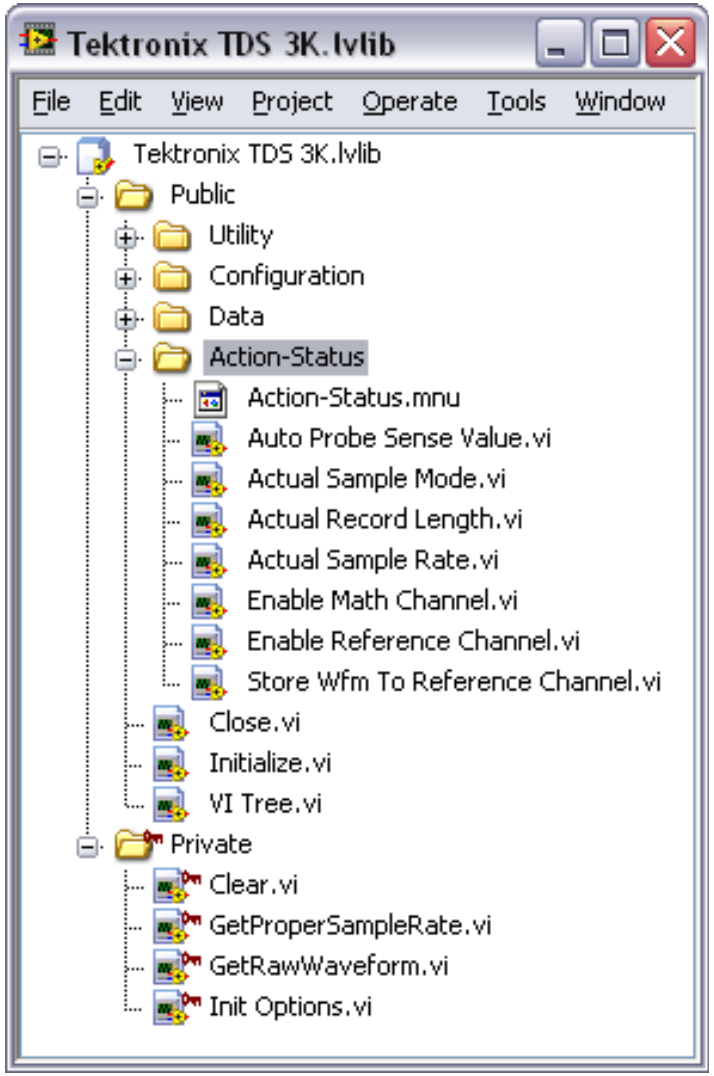
- Virtuelnim instrumentima i subVI
- Bibliotekama – Project Library (od LabVIEW 8)
- Klasama (od LabVIEW 8.20)

SubVI

- SubVI povećavaju čitljivost i laku izmenu kôda.
- SubVI se koristi kada:
  - Se isti kôd javlja više puta na blok dijagramu,
  - Blok dijagram postane složen, nepregledan i veliki,
  - Se u aplikaciji koriste Sequence strukture

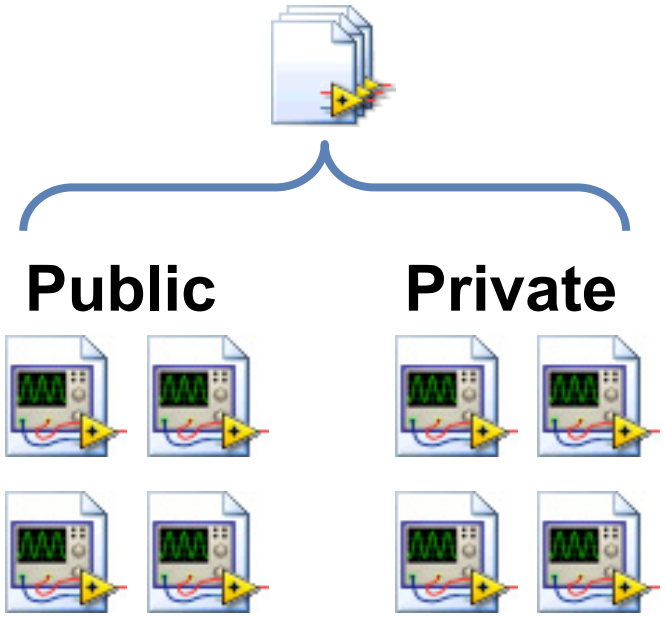
Biblioteke – Project Library

- Biblioteke su kolekcije povezanih virtuelnih instrumenta i ostalih pomoćnih fajlova (globalnih promenljivih, kontrola, itd.)
- Biblioteke su smeštene u tekstualne fajlove sa ekstenzijom *.lvlib

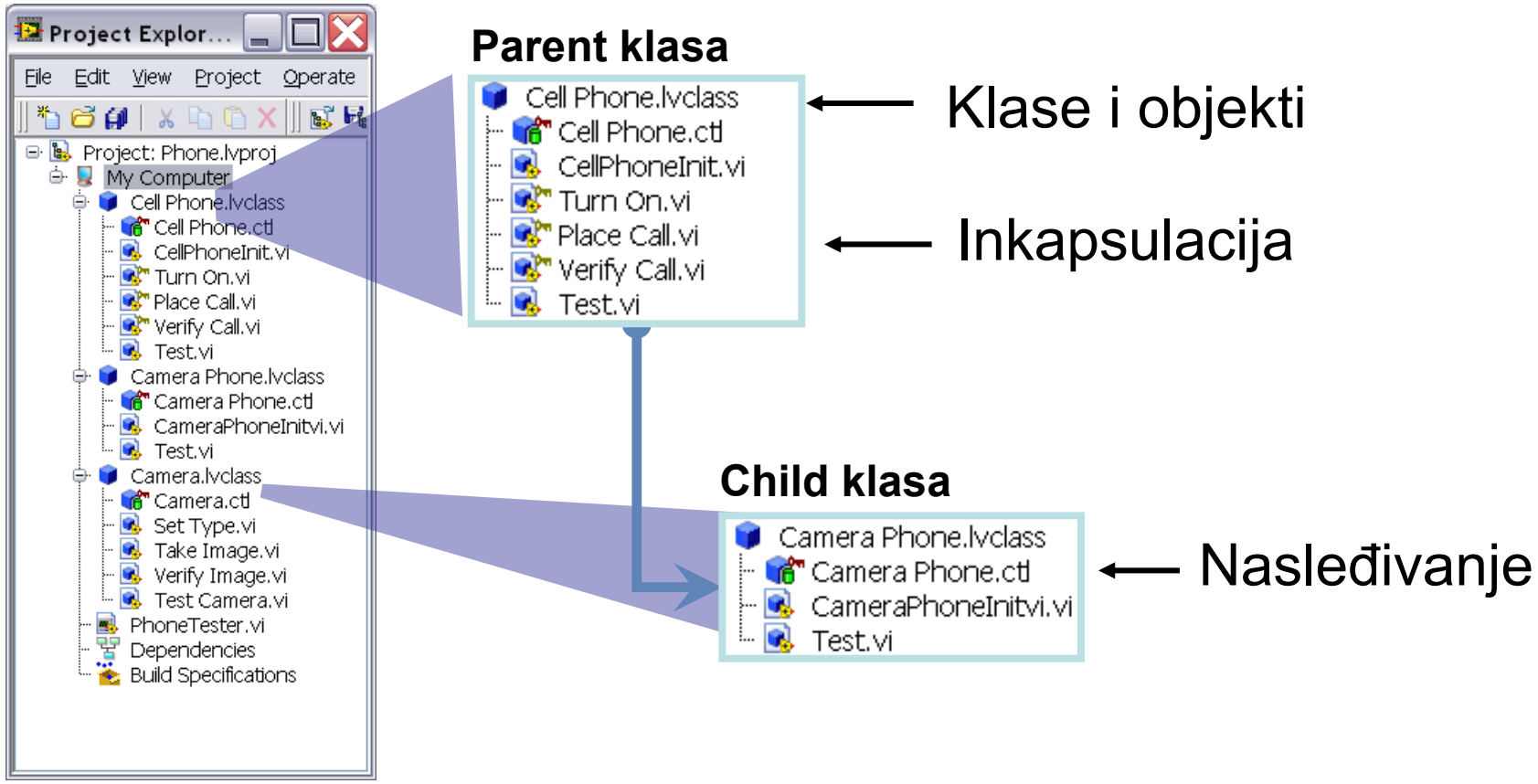


Prednosti biblioteka

- Smanjeni su konflikti sa imenima
  - Svaka biblioteka ima svoj **namespace** koji se odnosi na imena VI
- Restriktivan pristup virtuelnim instrumentima u biblioteci
  - Javni **– Public VIs**
  - Privatni **– Private VIs**
- Konzistentna ikona



Objektno orijentisano programiranje



Klase i objekti

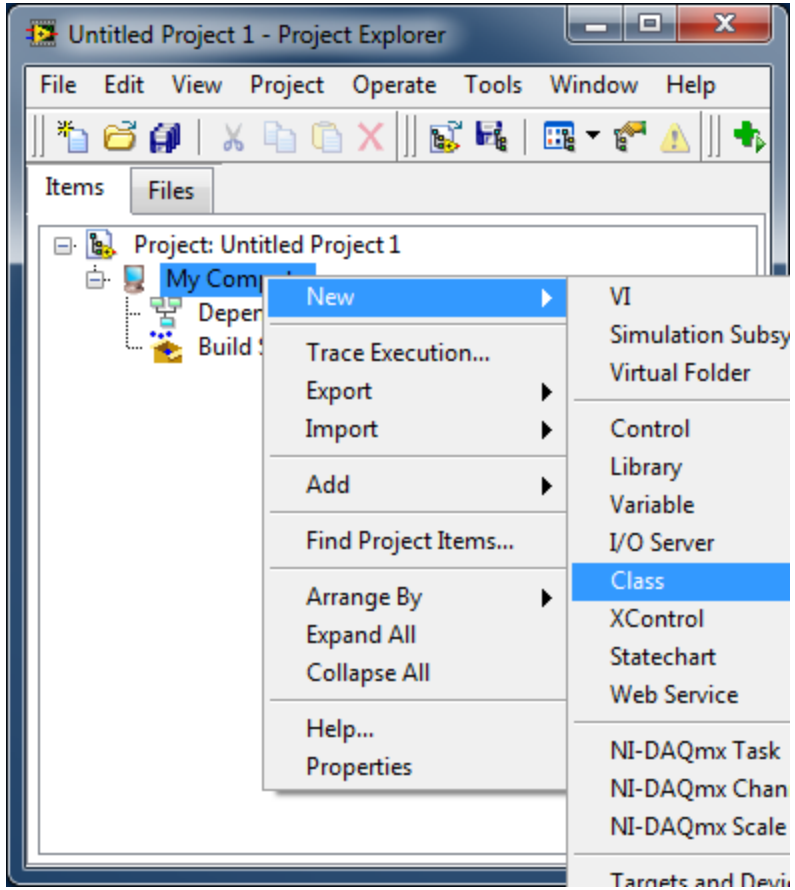
- Objekti su instance klase u aplikaciji
  - Odnose se na konkretne podatke
- Klasa određuje podatke u objektu i njegovo ponašanje
  - Svi objekti iste klase imaju iste osobine.

OOP – Testiranje mobilnog telefona

- Da bi testirali mobilni telefon i mobilni sa kamerom, potrebno je testirati mobilni telefon na pozive, a mobilne sa kamerom na pozive i fotografisanje
- Aplikacija se bazira na konkretne stvari i ***akcije***
  - Mobilni sa kamerama su vrsta mobilnih telefona koji imaju i kameru
  - Mobini telefoni ***primaju pozive***
  - Mobilni sa kamerama ***primaju pozive*** i ***snimaju fotografije***
  - Da bi ***testirali*** mobilni telefon, poziv mora biti ***primljen*** i ***verifikovan***
  - Da bi ***testirali*** mobilni sa kamerom, funkcija telefoniranja i kamere more biti ***testirana***
  - Da bi ***testirali*** kameru, fotografija mora biti napravljena i upoređena referentnom
- Stvari predstavljaju klase i objekte
- Akcije predstavljaju metode

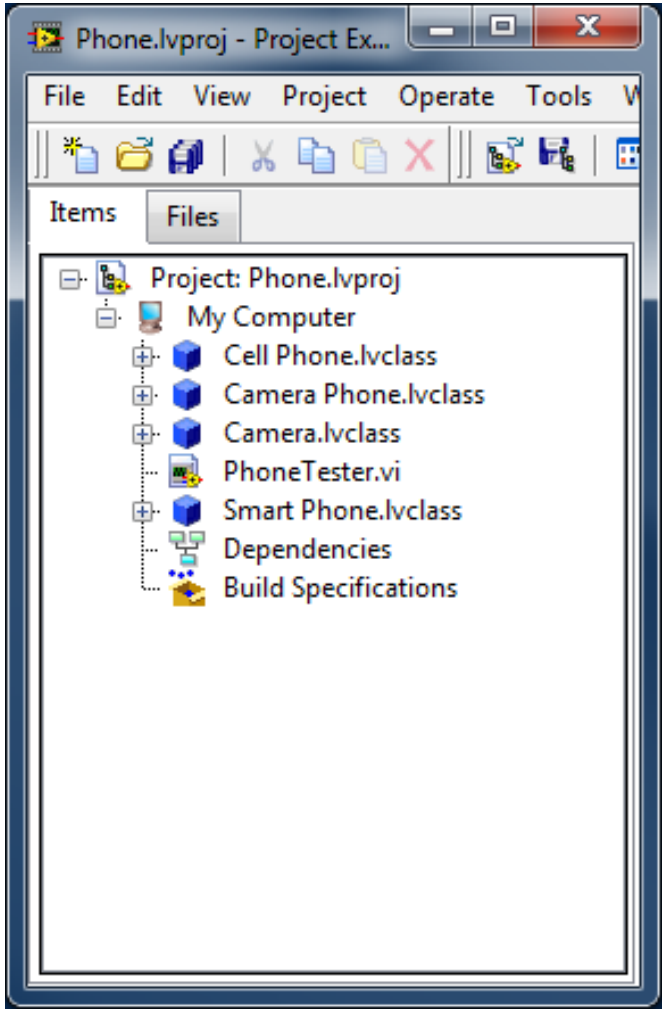
Kreiranje klasa u LabVIEW

- Kreirajte prazan projekt
- Dodajte klasu New>>Class
- Specificirajte privatne podatke klase u *.ctl
  - Definisanje klase efektivno definiše novi tip podataka
- Dodatne osobine klase
  - Ikona
  - Uzorak za VI ikonu
  - Boja veze



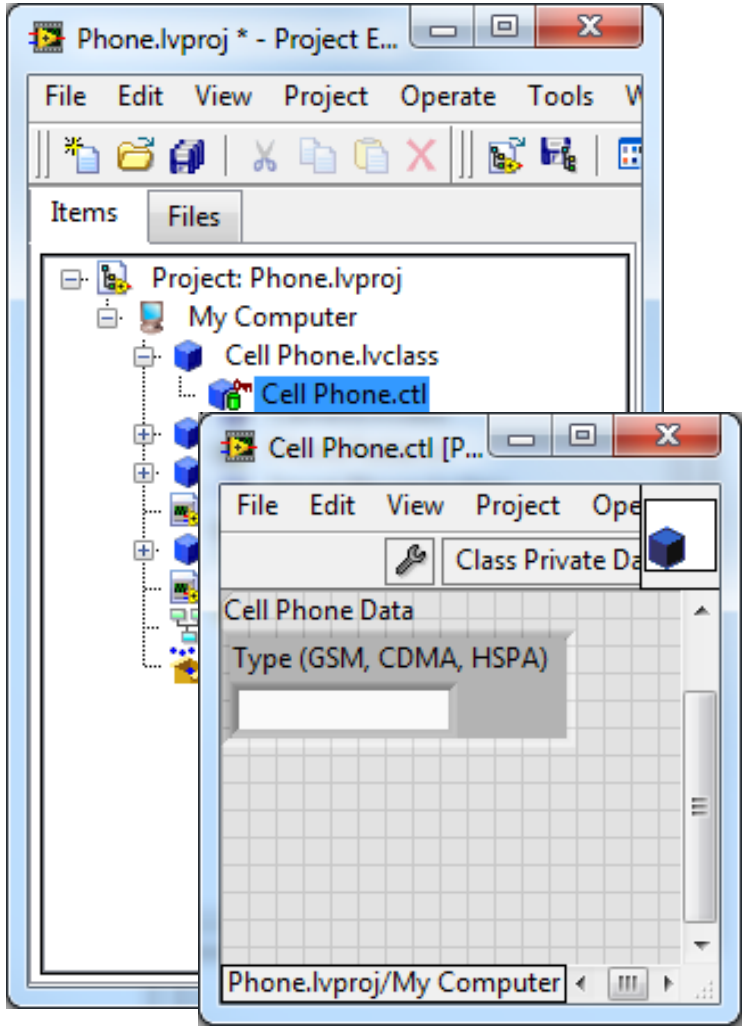
Klase u projektu

- Project explorer prikazuje klase kreirane u projektu
  - Cell Phone
  - Camera Phone
  - Camera
  - Smart Phone
- U okviru projekta je i virtualni instrument PhoneTester.vi

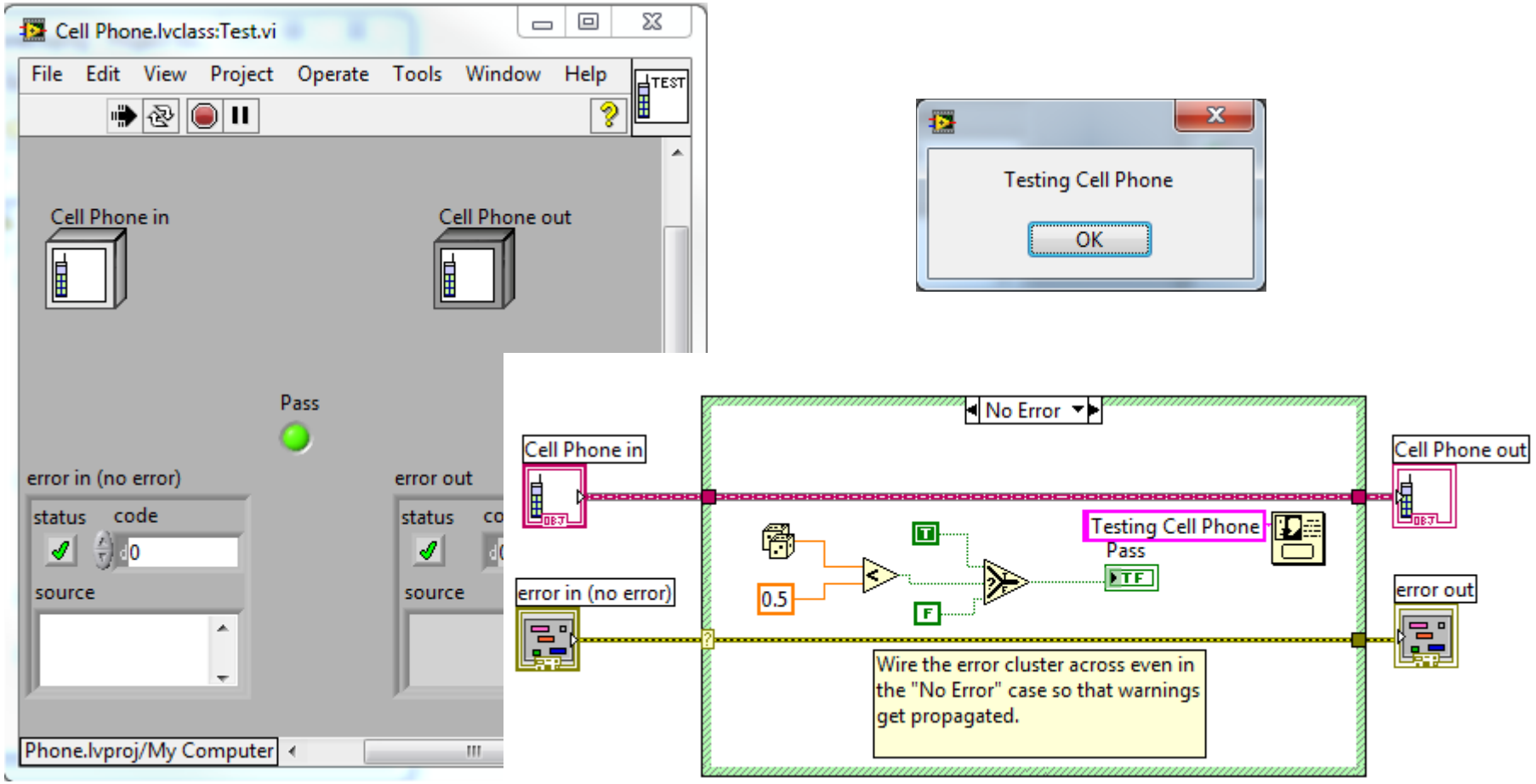


Privatni podaci klase (Private data)

- Klase
  - Cell Phone
  - Camera Phone
  - Camera
  - Smart
- Privatni podaci klase su definisani u pripadajućoj kontroli



Primer metoda – Test.vi



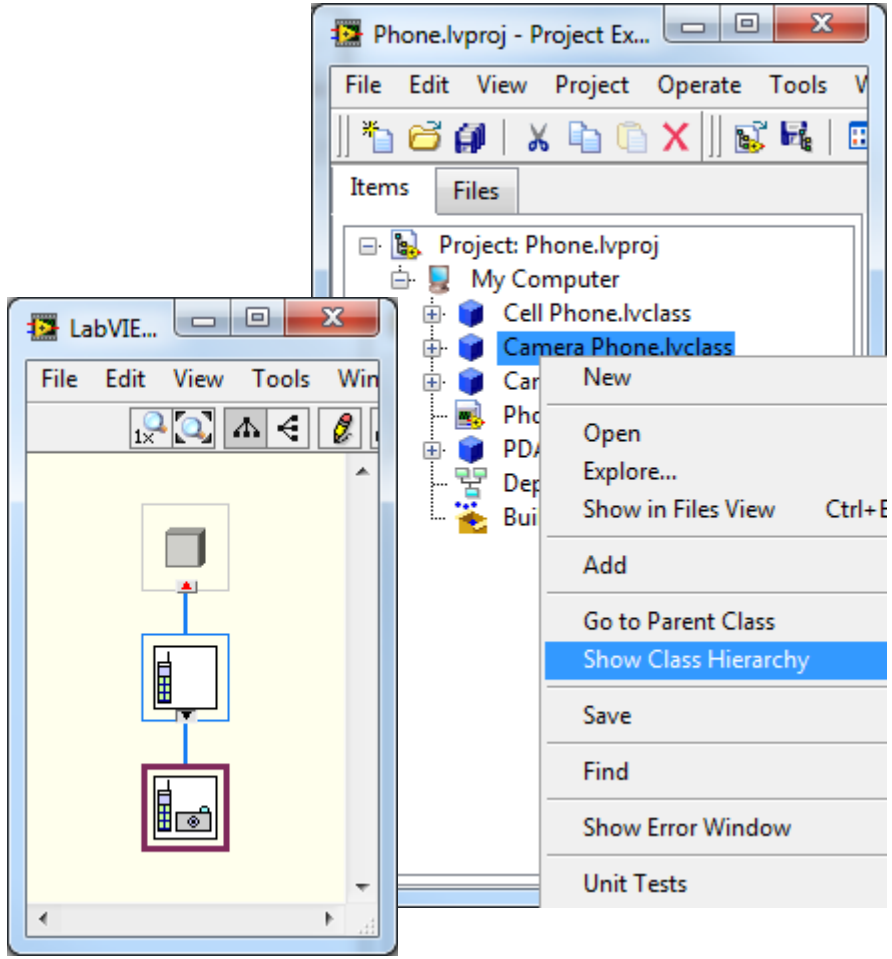
Nasleđivanje (Inheritance)

- Subklase izvedene iz klasa
- Uspostavlja „je“ relaciju između klasa
  - Primer: Mobilni sa kamerom „je“ mobilni telefon
  - Preuzima uspostavljenu funkcionalnost klase
- Specijalizacija
  - Proširuje ili potiskuje funkcionalnost u skladu sa specifičnim potrebama

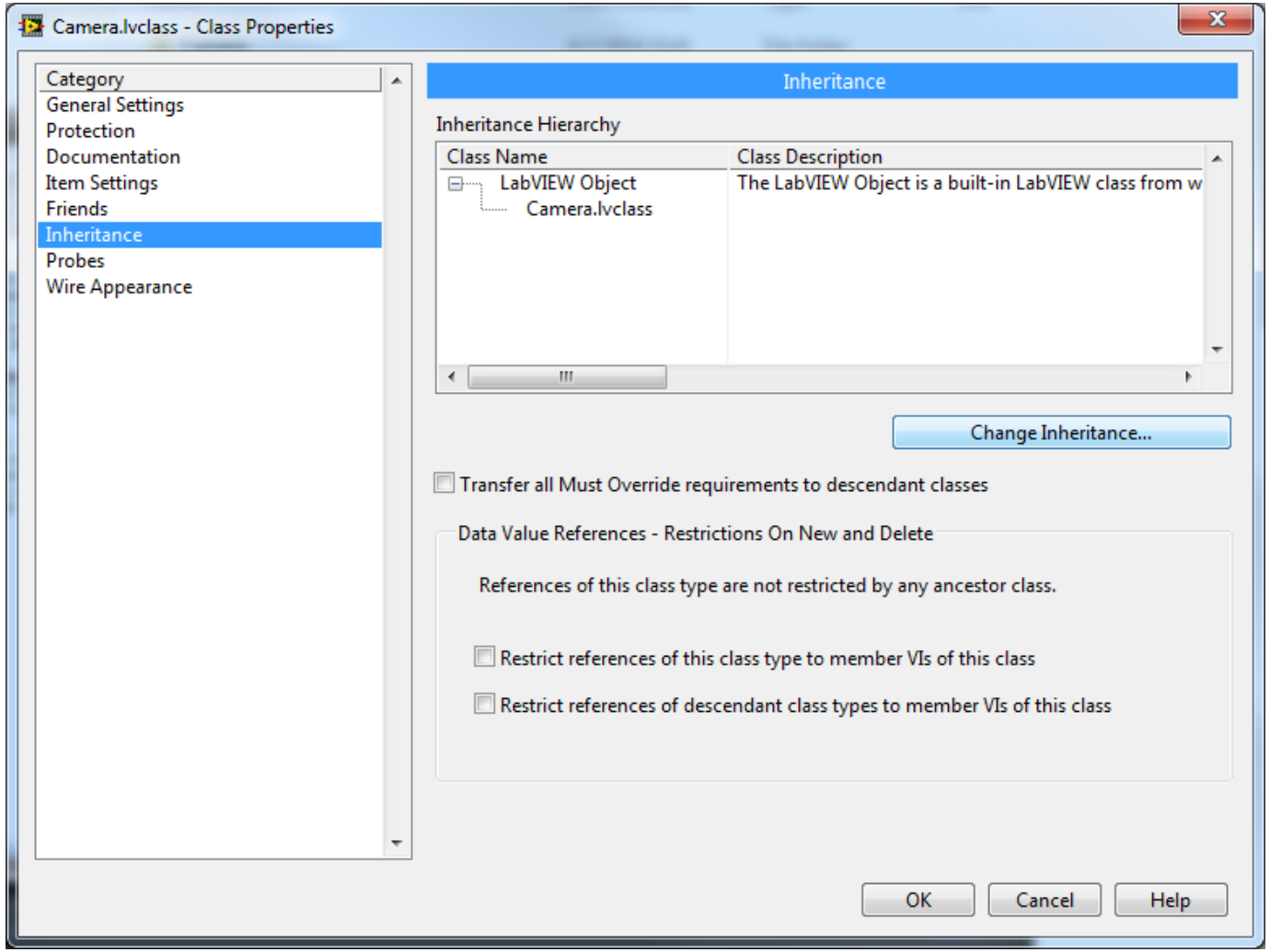
Testing Cell Phones – Inheritance

Klasa Camera Phone

- Nasleđena iz *Cell Phone* klase
- Podaci
  - Kamera
- Metodi
  - Test – proširuje “Cell Phone Class” Test.vi metod radi testiranja funkcionalnosti kamere



Nasleđivanje

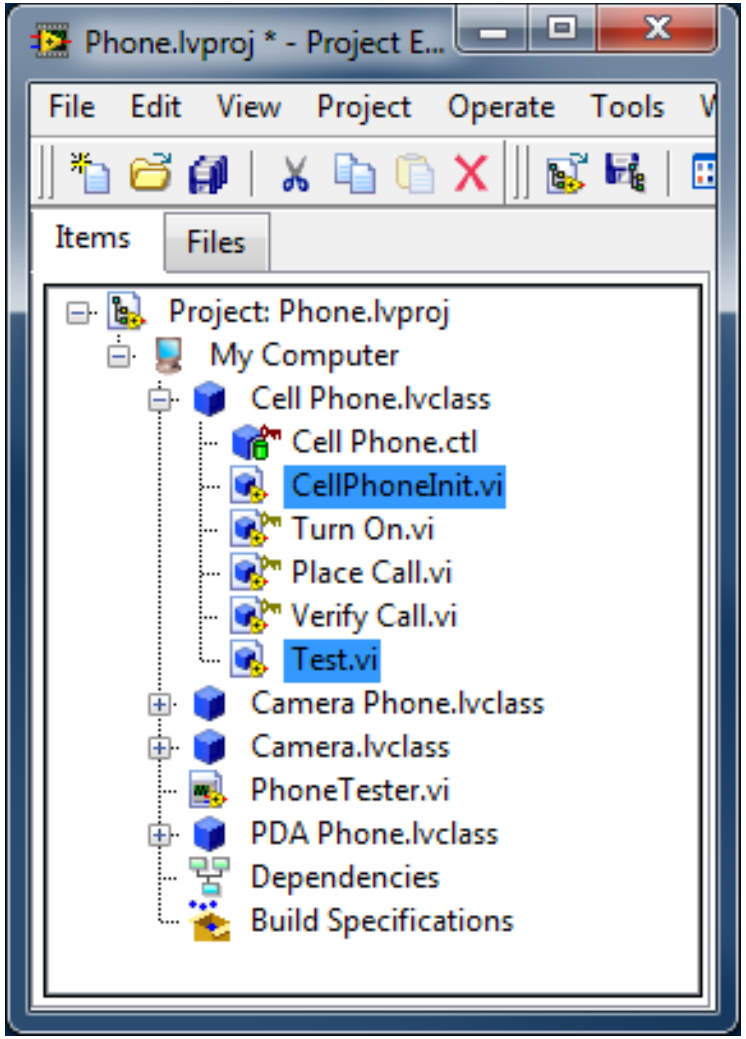


Inkapsulacija

- Tretira objekte kao crne kutije (blackbox)
  - Metodi i iterfejsi objekta su definisani
  - Interfejs se mora koristiti u aplikaciji
- Svi prodaci su private tipa
- Metodi mogu biti public, private, or protected

Inkapsulacija – metodi

- Top-level aplikacija jedino mora da inicijalizuje mobilni i testira ga
  - *CellPhoneInit* i *Test* metodi su javni

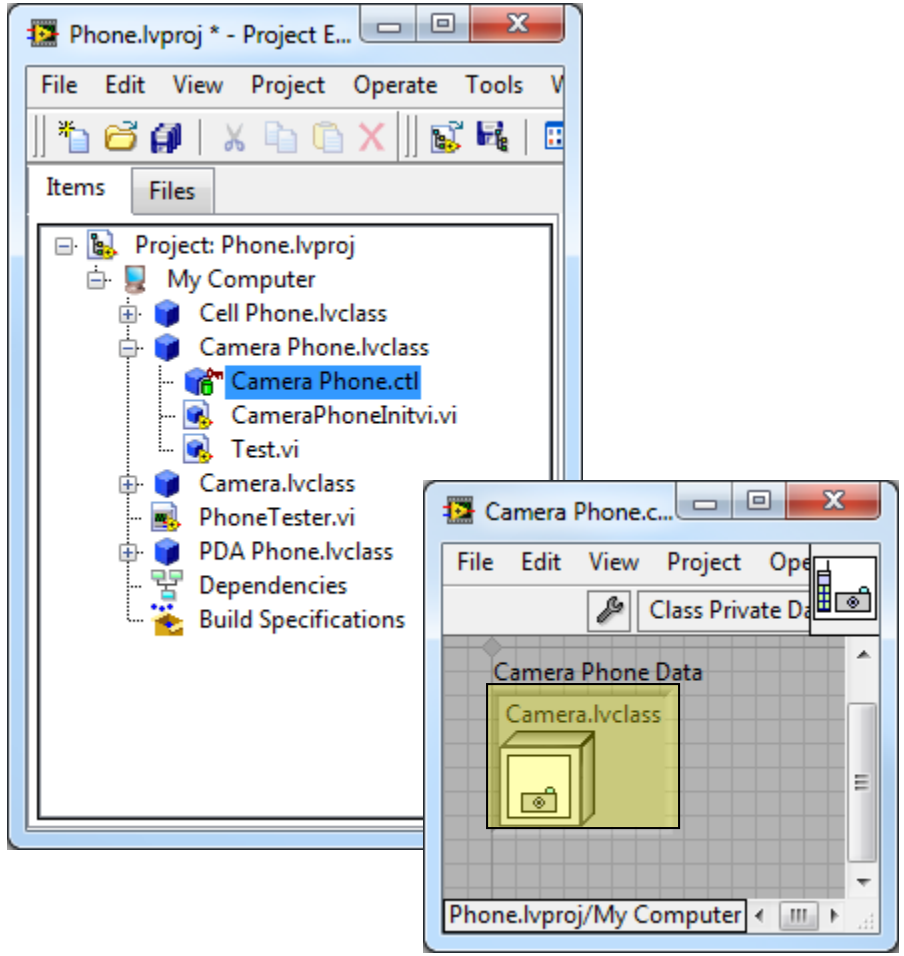


Struktura klase

- Definisanje klase kreira novi tip podataka
- Klase mogu sadržati druge klase kao *private* podatke
- Klasa Camera Phone sadrži klasu Camera

Struktura klase

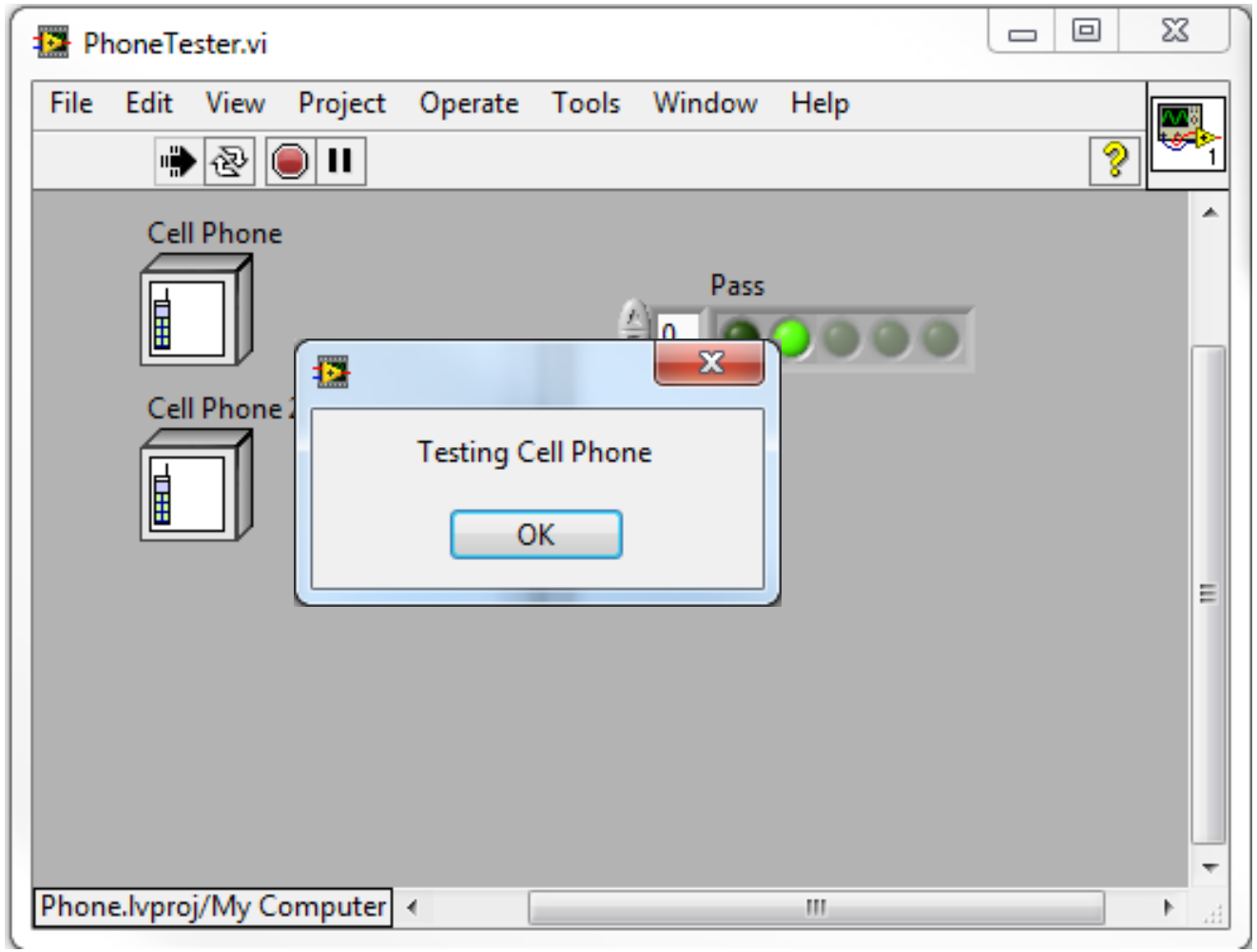
- Camera Phone klasa
  - *Nasleđena od Cell Phone*
  - Podaci
    - Camera Class
  - Metodi
    - Test (Cell Phone i Camera)
- Camera Class
  - Podaci
    - Camera Type
  - Metodi
    - Take Image
    - Verify Image
    - Test Camera



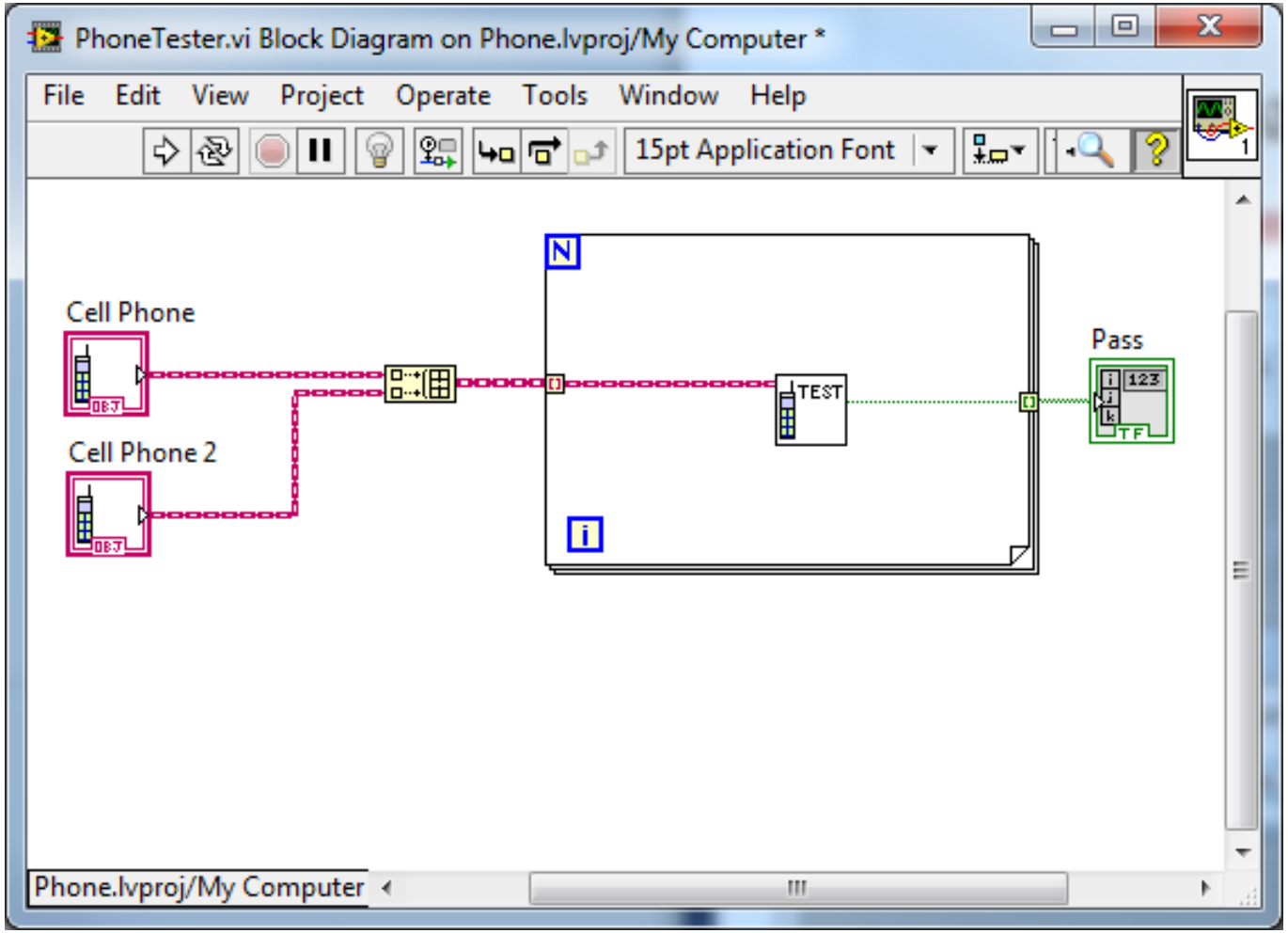
LabVIEW aplikacija sa klasama

- Pozivi objekata i metoda u blok dijagramu
- Smanjuje intervencije i izmene na kôdu korišćenjem nasleđivanja

Testiranje – poziv metoda Test.vi



Testiranje – poziv metoda Test.vi



Pregled

- Objektno orijentisano programiranje umanjuje potrebu za izmenom kôda i kompleksnost, pojednostavljuje proširenje funkcionalnosti aplikacija
- Osnovne paradigme su klase i objekti. Objekt predsavlja konkretnu instancu klase.
- Podaci u okviru klase su private tipa. Metodi mogu biti public, protected i private.
- Metod predstavlja akciju koju sprovodi objekat
- Nasleđivanje omogućava proširenje funkcionalnosti klase (objekta) bez ponovnog programiranja.