# Programarea orientată pe obiecte în Swift

**Programarea orientată pe obiecte este o paradigmă de programare fundamentală aplicabilă în Swift. "Spargerea" unei probleme în obiecte care apoi își trimit mesaje între ele  ar putea părea ciudată la început, dar este o abordare flexibilă pentru simplificarea sistemelor complexe**

**Obiectele pot fi folosite pentru a modela aproape orice:**

**- coordonatele pe o hartă,**

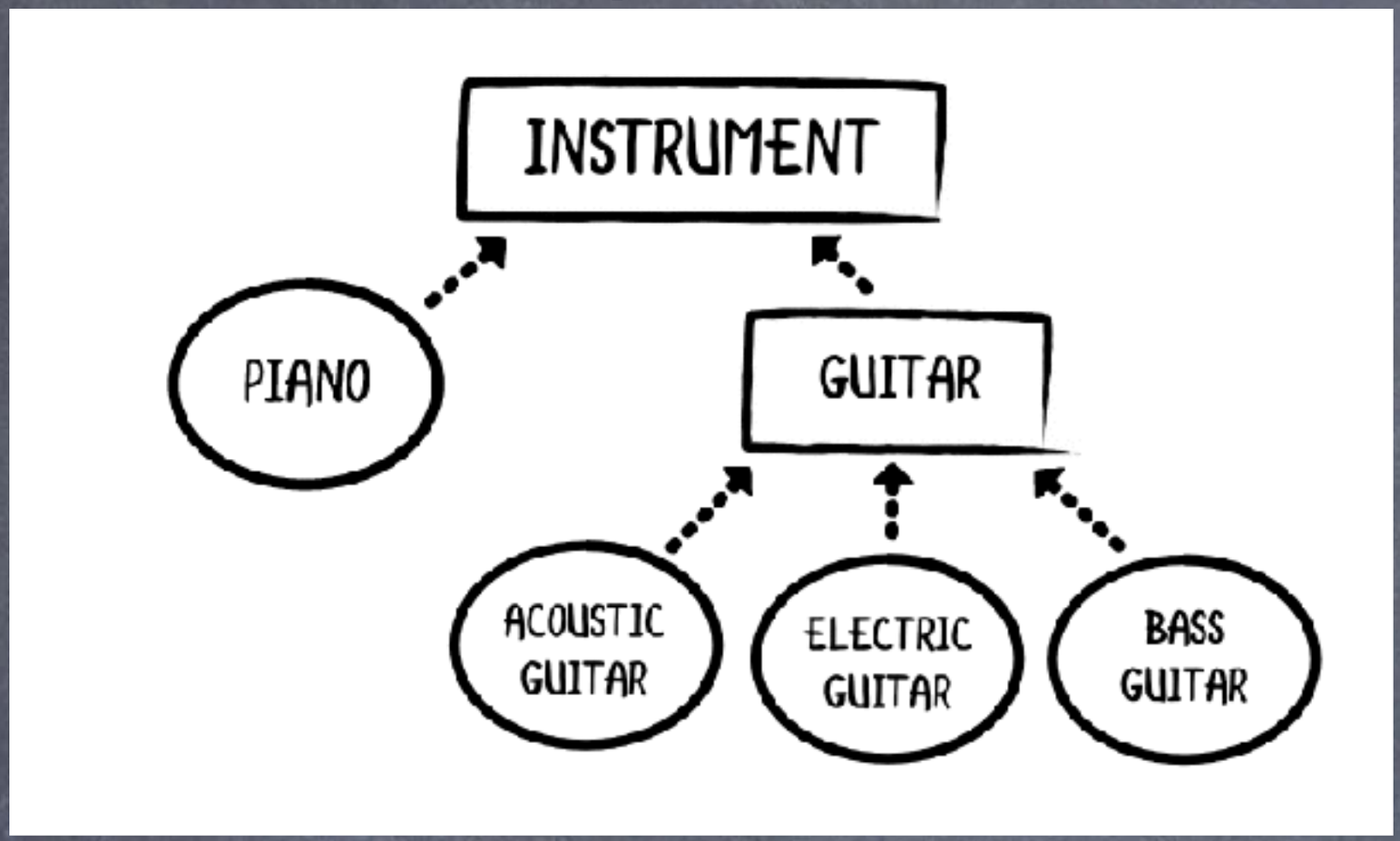
**- "atingerile" pe un ecran,**

**- modul în care  fluctuează ratele dobânzilor într-un cont bancar.**

Buzoianu Stefan
IOS Developer

# Programarea orientată pe obiecte în Swift

**Proiectarea aplicațiilor într-un mod orientat-obiect începe, de obicei, cu un concept general care se extinde ulterior la tipuri mai specifice.**




Buzoianu Stefan
IOS Developer

# Programarea orientată pe obiecte în Swift

## Proprietăți

```
// 1
class Instrument {
  // 2
        let brand: String
  // 3
  init(brand: String) {
    //4
    self.brand = brand
  }
}
```

```
// 1 clasa Instrument - clasa de baza
// 2 declarăm stored properties
// 3 initializer (constructor)
// 4 transfer de la parametru la stored property
```

Buzoianu Stefan
IOS Developer

# Programarea orientată pe obiecte în Swift

**Metode**

```
func tune() -> String {
  fatalError("Implement this method for \(brand)")
}
```

**Clasă abstractă? Metodă care nu face nimic?**

Buzoianu Stefan
IOS Developer

# Programarea orientată pe obiecte în Swift

## Clasa Music

```
final public class Music {
  public let notes: [String]

  public init(notes: [String]) {
    self.notes = notes
  }

  public func prepared() -> String {
    return notes.joined(separator: " ")
  }
}
```

## Clasa Instruments

```
open class Instrument {
        public let brand: String
        public init(brand: String) {
                self.brand = brand
        }

open func tune() -> String {
    fatalError("Implement this method for \(brand)")
  }
 open func play(_ music: Music) -> String {
    return music.prepared()
  }
final public func perform(_ music: Music) {
    print(tune())
    print(play(music))
  }
}
```

Buzoianu Stefan
IOS Developer

# Programarea orientată pe obiecte în Swift

## Moștenirea - Inheritance

```
// 1
class Piano: Instrument {
  let hasPedals: Bool
  static let whiteKeys = 52
  static let blackKeys = 36
  // 2
  init(brand: String, hasPedals: Bool = false) {
    self.hasPedals = hasPedals
    // 3
    super.init(brand: brand)
  }
  // 4
  override func tune() -> String {
    return "Piano standard tuning for \(brand)."
  }
  override func play(_ music: Music) -> String {
    // 5
    let preparedNotes = super.play(music)
    return "Piano playing \(preparedNotes)"
  }
}
```

1.Clasa Pian - subclasă a clasei părinte Instrument. Toate proprietățile și metodele stocate sunt în mod automat moștenite de clasa copil Piano și sunt disponibile pentru a fi folosite.

2. Initializer-ul construiește instanța cu o valoare default pentru hasPedals

3. După ce initializer-ul a setat stored property-ul hasPedals este apelat super.init Ce este un super.init? Ce face el de fapt?

4. "Suprascriem" metoda moștenită tune.  Ce se întâmplă de fapt?

5. "Suprascriem" și metoda play.  Cu ajutorul super apelăm metoda părinte pentru a obține notele muzicale. Ce facem cu ele?

Fiind derivată din clasa Instruments, ce moșteneste de la aceasta?

Buzoianu Stefan
IOS Developer

# Programarea orientată pe obiecte în Swift

## Method Overloading - cum adică supraîncărcare?

```
class Piano: Instrument {
.......
  override func play(_ music: Music) -> String {
   let preparedNotes = super.play(music)
    return "Piano playing \(preparedNotes)"
  }

 func play(_ music: Music, usingPedals: Bool) -> String {
    let preparedNotes = super.play(music)
    if hasPedals && usingPedals {
      return "Play piano notes \(preparedNotes) with pedals."
    }
    else {
      return "Play piano notes \(preparedNotes) without pedals."
    }
  }

}
```

Care dintre cele două metodele play este apleată de perform(_:) în acest caz?

Buzoianu Stefan
IOS Developer

# Programarea orientată pe obiecte în Swift

**Playground - cum putem pune în execuție cod Swift?**

```
let piano = Piano(brand: "Yamaha", hasPedals: true)

piano.tune()

let music = Music(notes: ["C", "G", "F"])

piano.play(music, usingPedals: false)

piano.play(music)

Piano.whiteKeys

Piano.blackKeys
```

Buzoianu Stefan
IOS Developer