انجمن جاواکاپ تقدیم می‌کند

دوره برنامه‌نویسی جاوا

برنامه‌نویسی شیءگرا
Object Oriented Programming (OOP)

الهام منظری

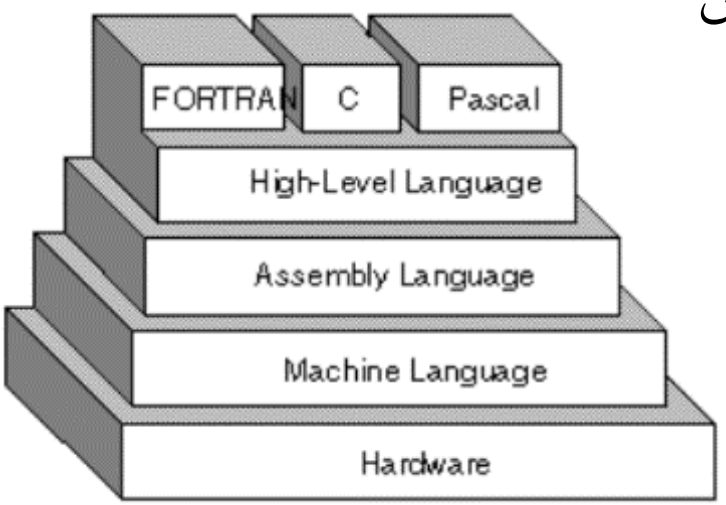
سرفصل مطالب

- برنامه‌نویسی شیءگرا
- Object Oriented Programming (OOP)
- کلاس، شیء، ویژگی‌ها و متدها
- مفهوم واسط (interface)
- مفهوم محصورسازی (Encapsulation)
- نوشتن و استفاده از اولین کلاس

رویکرد شیءگرا

پیشینه زبان‌های برنامه‌نویسی

- زبان ماشین
- اسمبلی: نامگذاری برای دستورات عددی زبان ماشین
- زبان‌های سطح بالا
  - مانند Fortran, Basic, C
  - ساختارهایی نزدیک‌تر به زبان انسان دارند
    - مثل حلقه، شرط و ...
  - از زبان ماشین و اسمبلی سطح‌بالاتر هستند
  - این رویکرد، پیشرفتی بزرگ بود
- اما همچنان برنامه‌نویس باید تحت ساختارهای کامپیوتر فکر کند
  - حتی با وجود زبان‌های سطح بالا
- بهتر است برنامه‌نویس به جای **ساختار کامپیوتر**، به **ساختار مسأله** فکر کند



فضای مسأله و فضای راه‌حل

- **برنامه‌نویسی** = ایجاد یک راه‌حل نرم‌افزاری برای یک موضوع واقعی
- مثال: نرم‌افزاری برای مدیریت کتاب‌ها و اعضا در یک کتابخانه
- **فضای مسأله** (Problem Space)
  - فضایی که در آن مسأله وجود دارد: مثل کتابخانه
  - مؤلفه‌های فضای مسأله‌ی کتابخانه: کتاب، عضو، قفسه، امانت و ...
- **فضای راه حل** (Solution Space)
  - فضایی که راه‌حل در آن ایجاد می‌شود: یک برنامه به زبان جاوا برای مدیریت کتابخانه
  - مؤلفه‌های فضای راه‌حل کتابخانه: پروژه، برنامه، متغیر، تابع
- **برنامه‌نویسی** = تلاش برای انجام نگاشت بین فضای مسأله و فضای راه‌حل

رویکرد شیءگرا

- به مسأله کتابخانه فکر کنید:
  - عناصر برنامه شما چه چیزهایی هستند؟
  - ما درباره توابع و متغیرها فکر می‌کنیم ...

اشیاء در مسأله کتابخانه:
یک کتاب،
یک عضو،
یک قفسه و ...

- **رویکرد شیءگرا** (Object Oriented) :
  - به برنامه‌نویس اجازه می‌دهد که عناصر فضای مسأله را نشان دهد
  - از مفاهیم و اصطلاحات همان مسأله در برنامه‌ای که می‌نویسد استفاده کند
- **شیء** (Object) :
  - موجودیت‌هایی که در فضای مسأله هستند
  - و در فضای راه حل (برنامه‌ها) هم دیده می‌شوند

برنامه‌نویسی شیءگرا

- Object Oriented Programming
  - برنامه‌هایی می‌نویسیم که با زبان فضای مسأله وفق پیدا می‌کنند
    - با کمک افزودن انواع جدید داده برای اشیاء همان مسأله
    - انواع داده محدود به آن‌چه زبان فراهم کرده، نیست
    - نوع داده «کتاب» و «قفسه» در کنار انواع ساده (مثل int و char)
  - وقتی برنامه را می‌خوانید، کلماتی می‌بینید که در مسأله معنا دارند
    - یک کتاب، یک عضو، امانت یک کتاب به یک عضو، و ...
  - این انتزاع، زبان را بسیار انعطاف‌پذیر و قوی می‌سازد

زبان‌های شیء گرا

- Smalltalk
  - یکی از اولین زبان‌های شیءگرای موفق
  - یکی از زبان‌هایی که الهام‌بخش جاوا بوده است
- C++
- C#
- **Java**

شیءگرایی در برابر رویکرد رویه‌ای

- عناصر برنامه‌نویسی شیءگر: اشیاء + تبادل پیام بین اشیاء
- عناصر برنامه‌نویسی رویه‌ای: توابع + متغیرها + فراخوانی تابع
- تفاوت به نحوه فکر کردن در زمان برنامه‌نویسی:
  - فکر کردن درباره اشیاء و رابطه بین اشیاء : روش شیءگرا
  - فکر کردن درباره حافظه، ساختار کامپیوتر و .. : روش رویه‌ای
  - تا قبل از این جلسه، عملاً با رویکرد رویه‌ای برنامه می‌نوشتیم

Procedural Programming and Object Oriented Programming

مشخصات برنامه‌نویسی شیءگرا

- «آلن کی» برنامه‌نویسی شیءگرا را در پنج ویژگی اصلی خلاصه می‌کند:

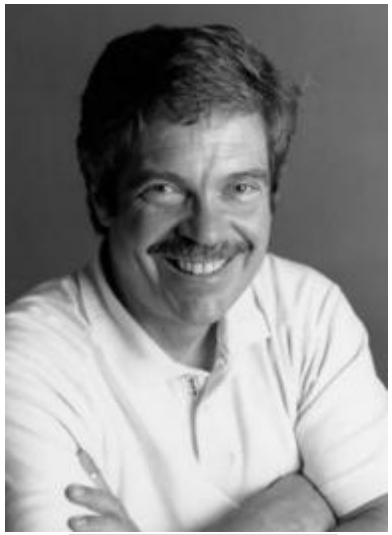
۱. هرچیز، یک شیء است

۲. یک برنامه مجموعه‌ای از اشیاء است
   - که با ارسال پیام به هم، به یکدیگر می‌گویند که چه کاری انجام دهند

۳. هر شیء، حافظه اختصاص یافته به خود را دارد
   - که از اشیاء دیگر ساخته شده است
   - (حالت شیء)

۴. هر شیء دارای یک نوع است

۵. همه اشیای از یک نوع خاص می‌توانند پیام‌های مشابه دریافت کنند

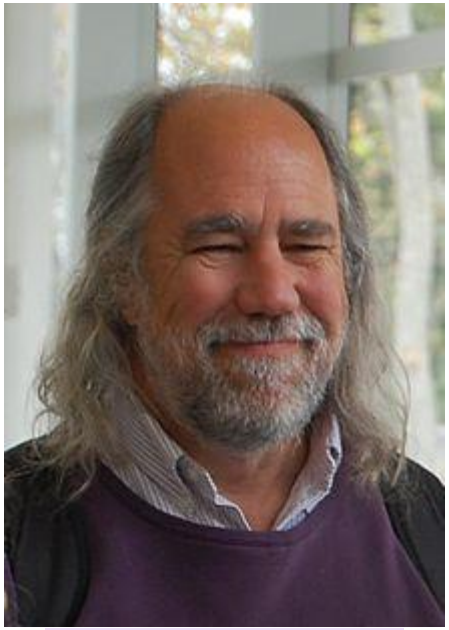


Alan Kay

توصیف بوچ از شیء

State
Behavior
Identity

- یک شیء، متشکل از **وضعیت**، **رفتار** و **هویت** است
- بوچ، «**هویت**» را به توصیف اشیاء اضافه کرده است
- هر شیء، **ویژگی‌هایی** دارد
  - که **وضعیت** (حالت) شیء را تعیین می‌کند
  - مثلاً: یک عضو کتابخانه، ویژگی‌هایی چون نام، سن و ... دارد
- هر شیء **متدهایی** دارد
  - **رفتارهایی** از خود نشان می‌دهد
  - مثلاً: رفتار «امانت گرفتن کتاب» برای شیء «عضو»
- و هر شیء، منحصر به فرد است: **هویتی** متمایز دارد
  - حتی اگر دو شیء «وضعیت» یکسان داشته باشند
  - **آدرس** منحصر به فرد در حافظه



Grady Booch

مفاهیم برنامه‌نویسی شیءگرا

شیء (Object)

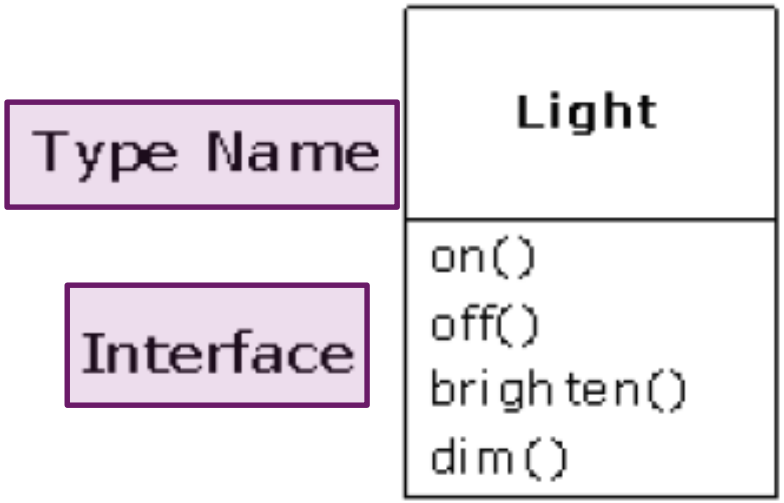
- موجودیت‌هایی که در فضای مسأله دیده می‌شوند
- کتابخانه:
  - کتاب «شاهنامه»، آقای «علی علوی» (عضو)
- بازی فوتبال:
  - علی کریمی، فرهاد مجیدی، ورزشگاه آزادی، توپ
- برنامه بانکی:
  - مریم علوی (مشتری)، شعبه آزادی

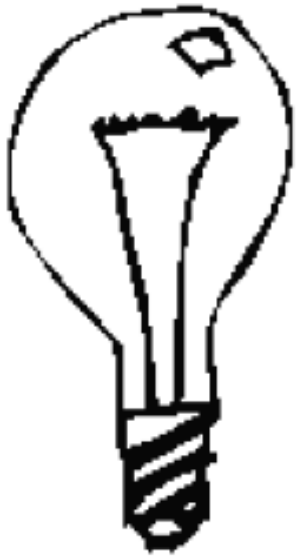
کلاس (Class)

- نوع، رده، یا دسته‌ای از اشیاء که «ویژگی‌ها» و «رفتار» مشابه دارند
- هر کلاس، نمونه‌های مختلفی (اشیاء) دارند
- کتابخانه: کتاب، عضو، قفسه
- بازی فوتبال: بازیکن، تیم، ورزشگاه
- برنامه بانکی: مشتری، شعبه، حساب
- هر کلاس، رفتار (Behavior) و ویژگی‌هایی تعریف می‌کند (Property)
  - هر شیء از این کلاس دارای این ویژگی‌ها و رفتارهاست
- مثال: عضو کتابخانه
  - ویژگی‌ها: نام، سن، شغل
  - رفتارها: امانت گرفتن کتاب

نمایش یک لامپ حبابی

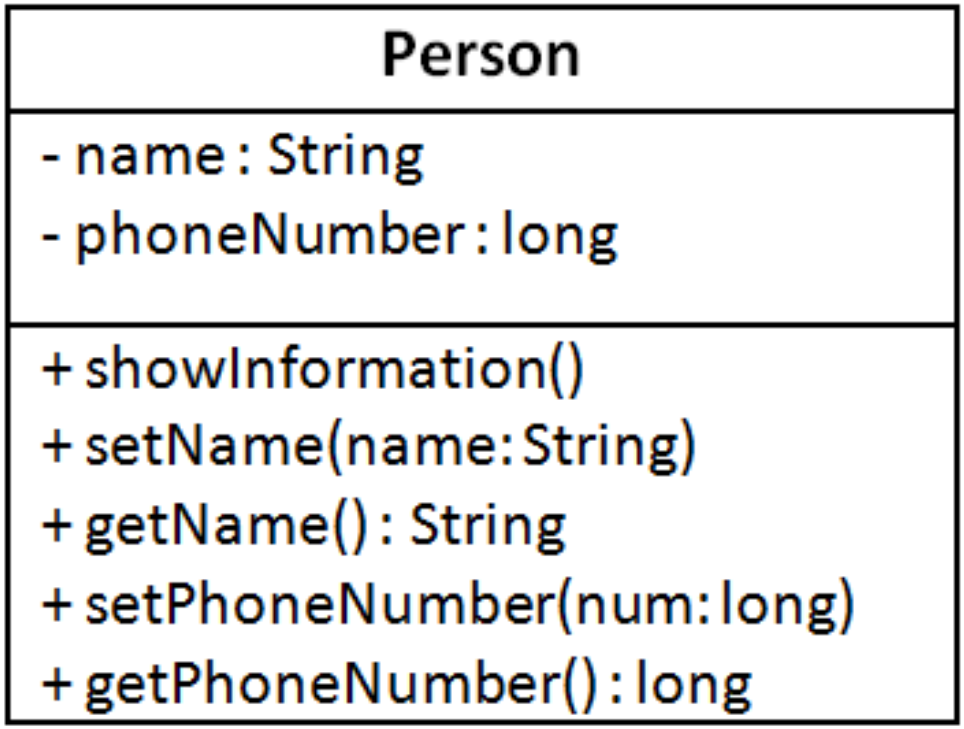
UML Class Diagram





```
Light lt = new Light();
lt.on();
```

شخص در یک سیستم آموزشی



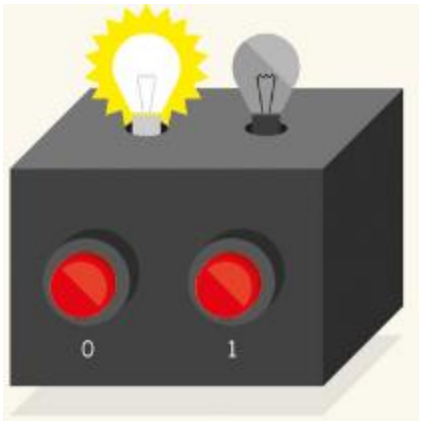

```
Person person1 = new Person();
person1.setName("Taghi Taghavi");
person1.setPhoneNumber(66166601L);
person1.showInformation();
```

کنترل دسترسی

- در طراحی کلاس‌ها: می‌توانیم دسترسی به بخش‌هایی از کلاس را محدود کنیم
- بخش‌های **عمومی** (**Public**) و **خصوصی** (**Private**) در کلاس
- استفاده‌کننده از کلاس فقط می‌تواند از بخش‌های عمومی آن استفاده کند
- منطقه عمومی واسط کلاس (interface) را نشان می‌دهد
- بخش‌های خصوصی، داخل همان کلاس قابل استفاده هستند
  - ولی توسط کلاس‌های دیگر، لزوماً قابل استفاده نیستند
- بخش‌های خصوصی، از دید استفاده‌کنندگان پنهان است
  - پیاده‌سازی پنهان (implementation hiding)

محصورسازی (Encapsulation)

- محصورسازی: **تعریف ویژگی‌ها و رفتارها** + **پنهان‌سازی پیاده‌سازی**
  - لفاف‌بندی یک مفهوم در فضای مسأله در قالب یک کلاس از اشیاء
- تعریف کلاس‌هایی جدید با **واسط‌های عمومی** و **پیاده‌سازی‌های پنهان**
- محصورسازی: تعریف نوع (کلاس) + تعریف واسط کلاس (نحوه استفاده از کلاس)
- محصول (کلاس) مثل یک جعبه سیاه (Black Box) خواهد بود
  - که فقط به شکل خاصی از آن می‌توان استفاده کرد
- بسیاری از محصولات تجاری محصورسازی شده‌اند
  - تلویزیون، تلفن همراه، ...



چرا محصورسازی؟

- بهره‌برداری و استفاده ساده‌تر
  - استفاده‌کننده درگیر جزئیات پیاده‌سازی نمی‌شود
- پیاده‌سازی باز ممکن است به استفاده اشتباه منجر شود
  - پنهان‌سازی پیاده‌سازی، بروز خطا را کاهش می‌دهد
- این نحوه طراحی، زیباتر است

Implementation

Interface

واسط (Interface)

- هر شیء می‌تواند رفتارهای خاصی از خود نشان دهد
  - درخواست‌های خاصی را پاسخ دهد
- **واسط شیء:**
  - مجموعه رفتارهایی که برای یک شیء قابل فراخوانی است
- مثلاً «امانت داده شدن»، بخشی از واسط کلاس کتاب است
  - هر کتاب رفتار امانت داده شدن دارد
  - مثال: رفتار «امانت داده شدن» برای کتاب «شاهنامه» فراخوانی می‌شود
- واسط هر شیء در کلاس (نوع) شیء مشخص می‌شود
- نکته: interface یک معنای خاص هم در جاوا دارد (بعداً می‌بینیم)

تعریف اولین کلاس

نحوه تعریف کلاس

- تعیین نام کلاس
- تشخیص ویژگی‌ها (Property)
  - هر شیء از این کلاس، چه ویژگی‌هایی دارد؟
  - هر ویژگی را به صورت یک متغیر کلاس تعریف می‌کنیم
- تشخیص کارکردها (Method)
  - هر شیء از این کلاس، چه کارکردهایی دارد؟
    - چه عمل‌هایی انجام می‌دهد؟ چه پیام‌هایی را می‌پذیرد؟
  - هر کارکرد را به صورت یک متد در کلاس تعریف می‌کنیم

مثال: مستطیل

- مثال: می‌خواهیم نوع موجودیت **مستطیل** را محصورسازی کنیم
- هر مستطیل چیست؟
  - یک شیء
  - که دارای طول و عرض است (ویژگی‌ها)
  - امکان تعیین طول و عرض دارد (رفتار)
  - امکان محاسبه محیط و مساحت دارد (رفتار)

تعریف کلاس مستطیل

```
public class Rectangle {
    private int width, length;
    public void setWidth(int w) {
        width = w;
    }
    public void setLength(int l) {
        length = l;
    }
    public int calculateArea(){
        return width*length;
    }
    public int calculatePerimeter(){
        return (width+length)*2;
    }
}
```

منطقه خصوصی:
پیاده سازی پنهانی

منطقه عمومی:
واسط (interface)

چگونه از کلاس مستطیل استفاده کنیم؟

شیء (Object)

```
Rectangle rect = new Rectangle();
rect.setWidth(2);
rect.setLength(7);
System.out.println(rect.calculateArea());
System.out.println(rect.calculatePerimeter());
```

Object creation or instantiation

ساخت شیء یا نمونه‌سازی

Method Invocation (Message Passing)

فراخوانی متد (ارسال پیام)

کوییز

کوییز ۱

- فرض کنید می‌خواهیم برنامه «بازی فوتبال» بنویسیم.
- هر یک از این موارد، با کدام‌یک از مفاهیم شیءگرا منطبق است؟
- (کلاس، شیء، متد، ویژگی و ...)

فوتبالیست	کلاس
علی دایی	شیء
شوت زدن	متد
سن بازیکن	ویژگی

تمرین عملی

تمرین: مسأله بازی فوتبال

- مفاهیمی مانند موارد زیر را در یک برنامه تمرین کنید

- فوتبالیست
- علی دایی
- شوت زدن
- سن بازیکن

- نمونه‌سازی
- واسط
- بخش عمومی و بخش خصوصی

- نگاهی به برنامه: نزدیک شدن فضای مسأله و راه‌حل

جمع‌بندی

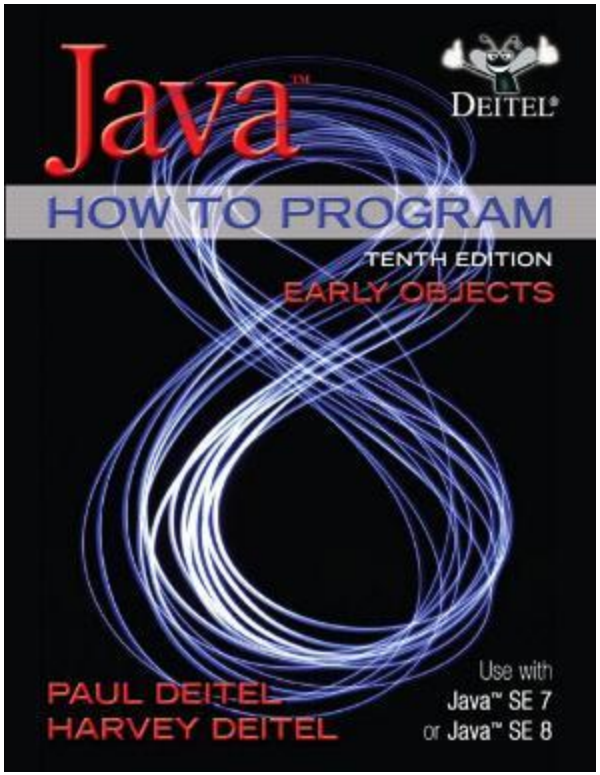
جمع‌بندی

- کلاس (Class)
- شیء (Object)
- نمونه‌سازی (Instantiation)
- رفتارهای اشیاء در قالب متدها (Method)
- ویژگی‌های اشیاء (Property)
- محصورسازی
- واسط

مطالعه کنید

- فصل‌های اول و سوم کتاب دایتل

Java How to Program (Deitel & Deitel)



1- Introduction
3- Introduction to Classes, Objects, Methods and Strings

- تمرین‌های همین فصل‌ها از کتاب دایتل

تمرین

- با کمک زبان جاوا، برای هر یک از این موارد کلاسی تعریف کنید. از هر کلاس نمونه‌هایی (شیء) بسازید. ویژگی‌های هر شیء را تعیین کنید. رفتارهای هر شیء را فراخوانی کنید.

- کتاب
- عضو کتابخانه

- مشتری
- شعبه
- حساب
- کارت بانکی

جستجو کنید و بخوانید

- کلمات و عبارات پیشنهادی برای جستجو:

- Object Oriented Programing
- Procedural Programming
- Interface
- Encapsulation
- UML Class Diagram

پایان