入门与编程技能训练

如何自学Java？

北京理工大学计算机学院
金旭亮

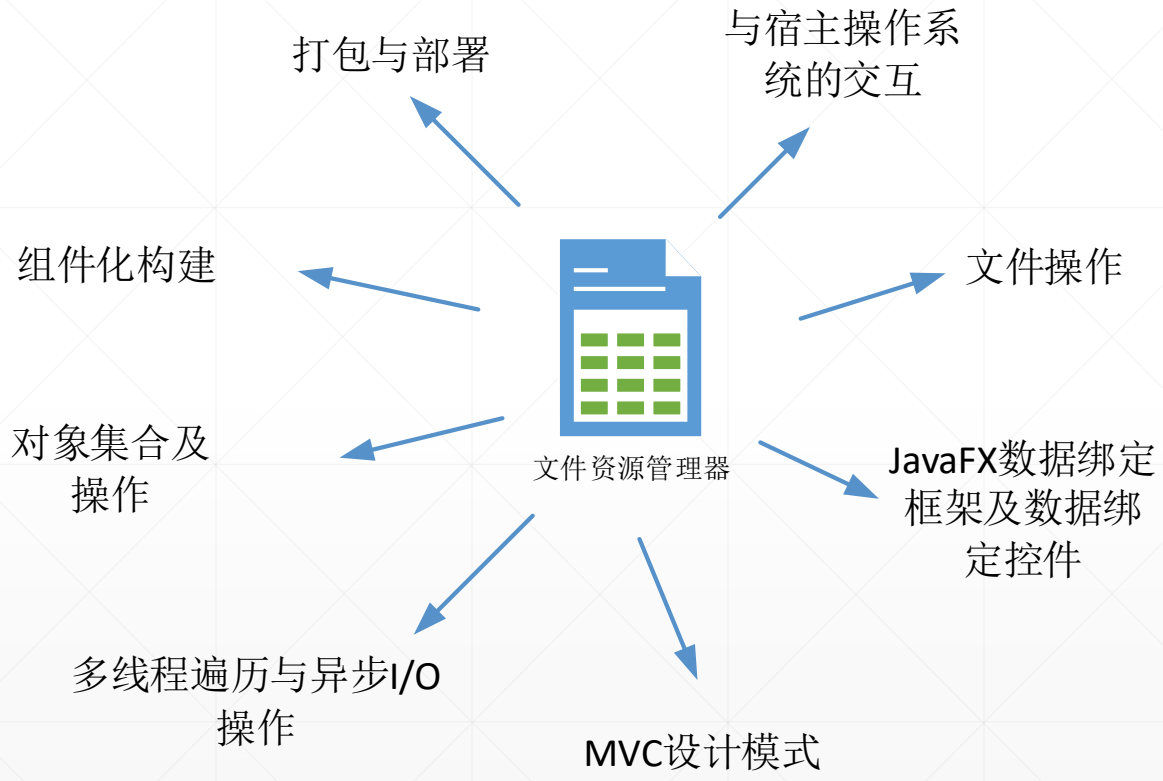
2017.10.11 知乎Live

1 Java学习之“武林秘籍”

学习Java要完成的任务

1. 理解与把握软件开发相关的**计算机科学理论**
2. 能使用各种**Java开发框架与工具**开发软件系统
3. 通过刻意训练培养出**通用的职业软件开发技能**
4. 将想法与念头变为现实，**创新创造。**

“项目驱动式”的学习方法



在掌握了Java基础知识与编程技巧之后，就可以采用这种学习方法。

以完成特定项目为中心，“缺哪补哪”、“哪个不会学哪个”。

学习的成果与进展，体现为一个个的具体项目，一步一个脚印。

项目的选择原则：
由简单到复杂，循序渐进，遵循刻意练习的基本原则，每个项目都有**新东西**要学。

四则运算计算器

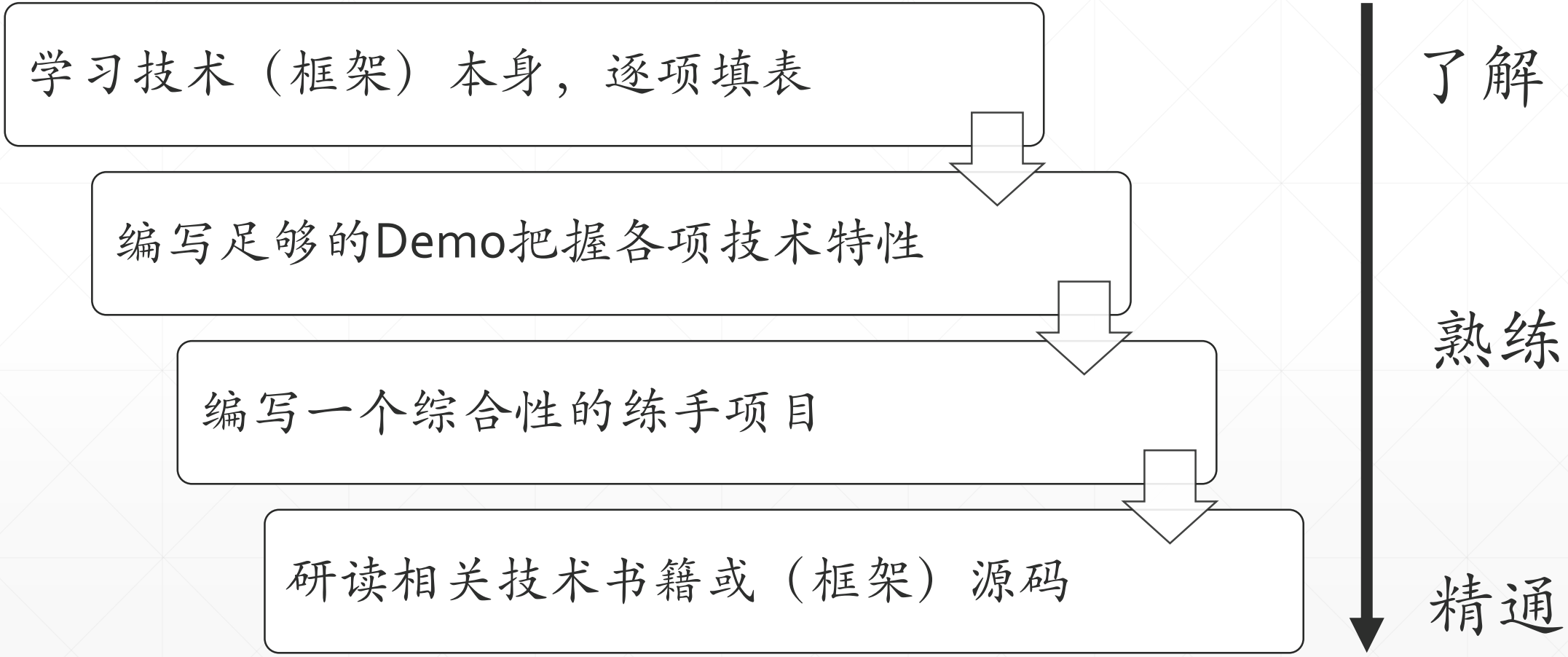
- 1 黑底白字的控制台版计算器
  - 掌握四则运算表达式解析的数据结构与算法
- 2 有窗体的桌面单机版计算器
  - 掌握JavaFX开发技术
- 3 使用Spring MVC的Web版计算器
  - 学会Web开发技术
- 4 Android版计算器
  - 掌握手机开发技术

围绕着“计算器”进行的刻意训练

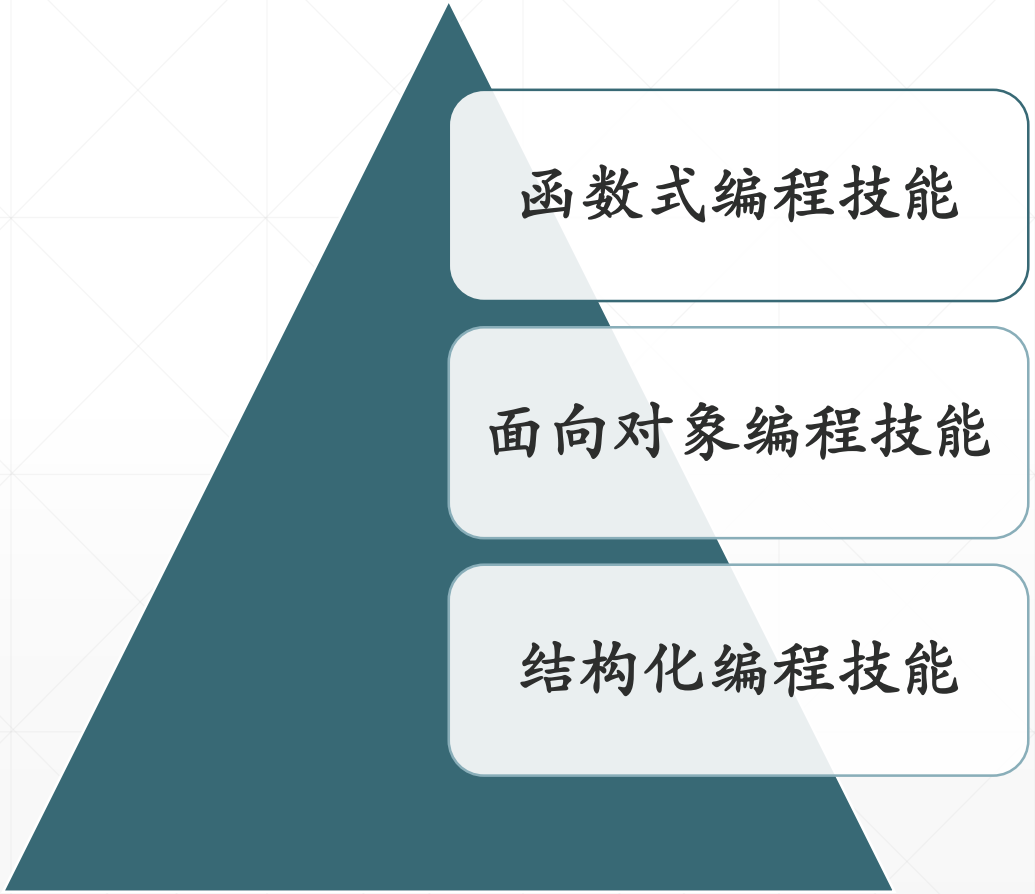
Java特定领域技术（框架）学习模板

学习项目	说明
应用场景	这个技术用在哪?
技术特性	这个技术有哪些特点？与其他的类似技术相比，优缺点在哪？.....
关联技术与知识网络	这个技术与哪些技术紧密相关？它包容哪些技术点，这些技术点之间又有哪些关联？.....
使用方法与步骤	在实际开发中，应用这个技术需要进行哪些配置，需要注意哪些地方，具体步骤如何？.....
编程模型与技术要点	这个技术包容哪些核心组件，这些组件各自的职责如何，它们是如何相互协作的? 这个技术有哪些技术要点？针对每个要点编写小的Demo，将相应成果整理到个人资料库或代码库中。
核心开发场景的技术解决方案	针对特定场景所面临的实际问题，这个技术是怎么解决的？有哪些成熟的解决方案?

软件技术（框架）的总体学习顺序



Java程序员必须具备的三种职业开发技能

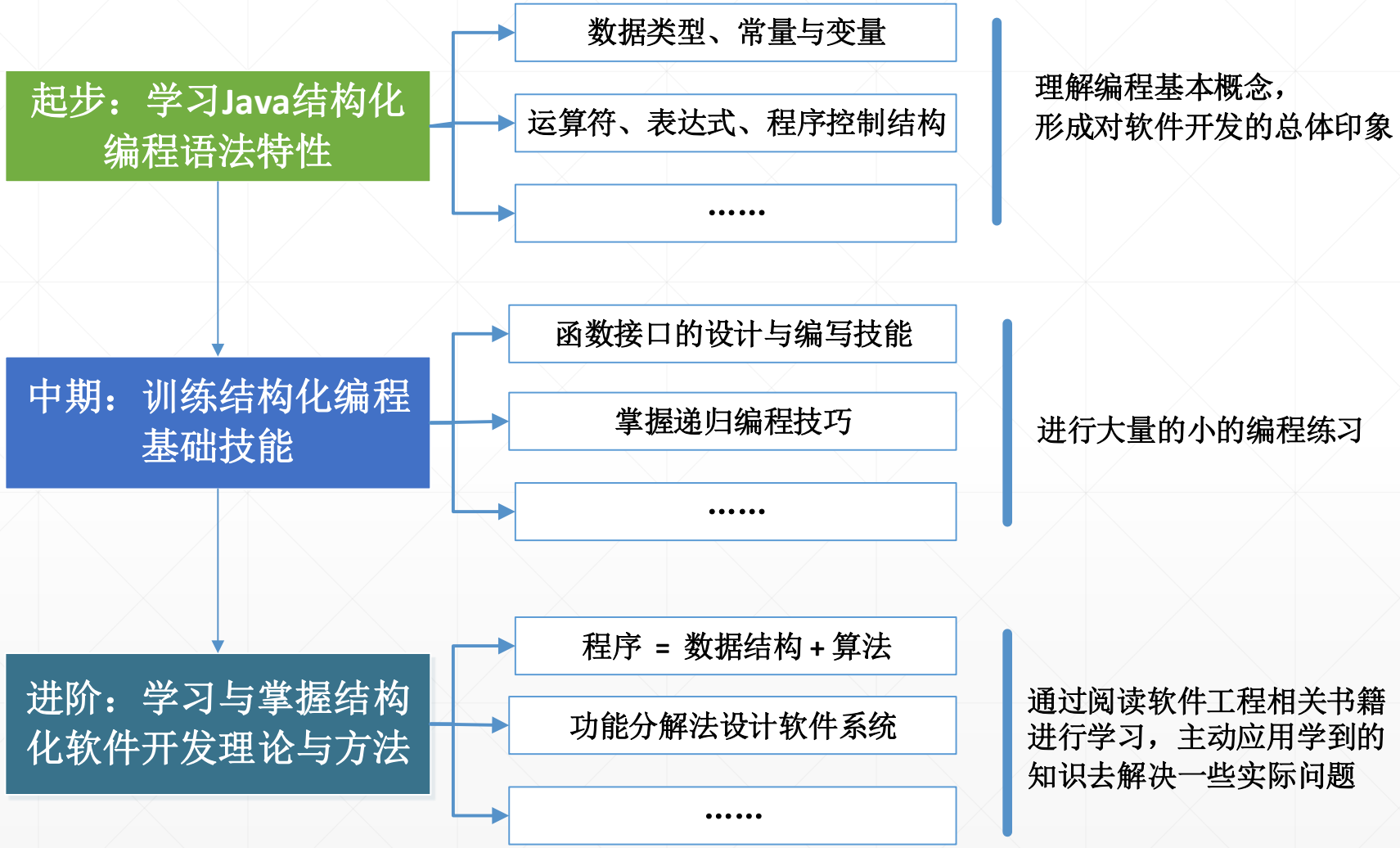


日益流行，有效解决现代软件系统的开发困境，成长为中高级程序员必备。

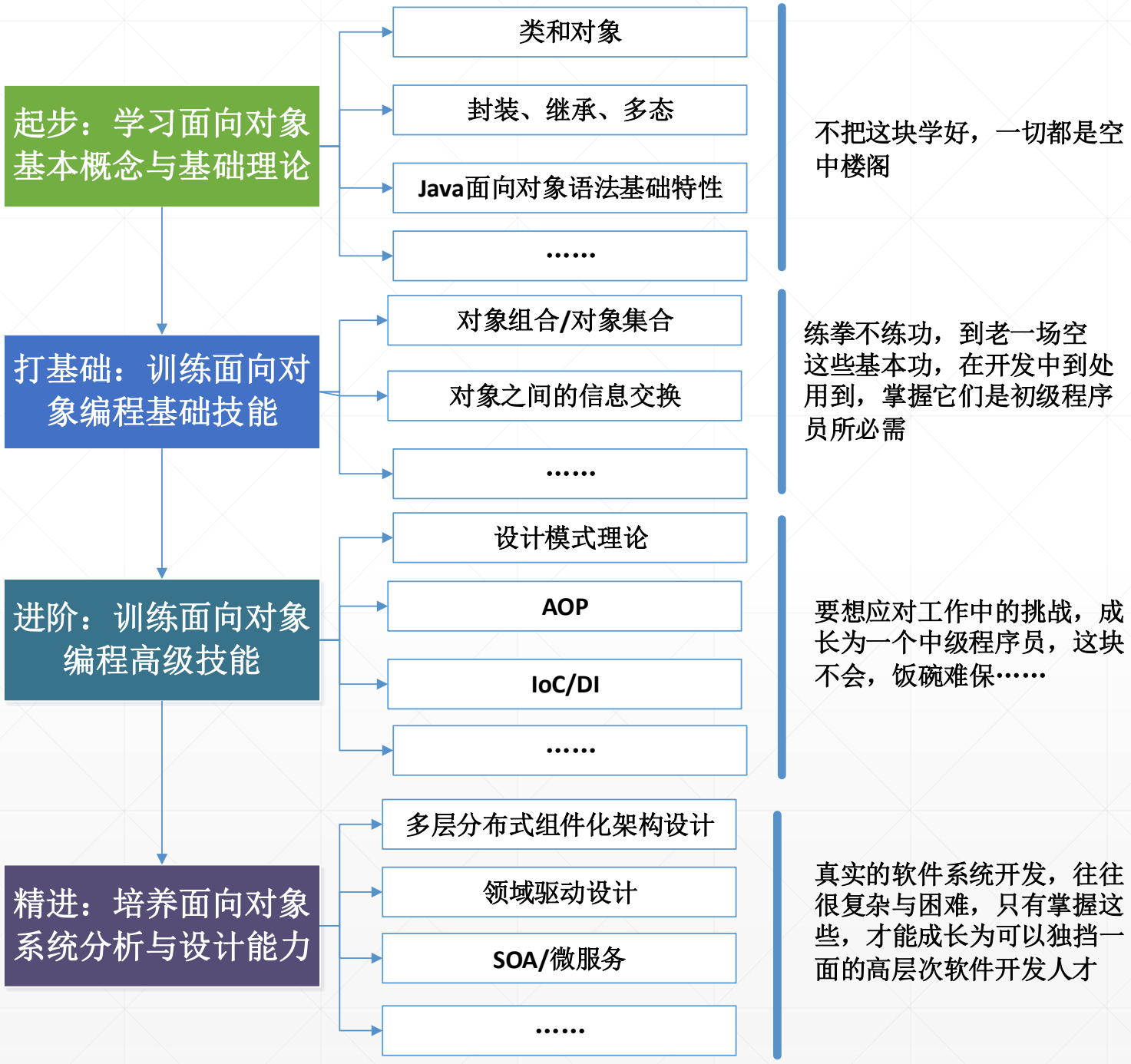
当前的主流，完成日常工作所必需，初学者需要花费大量时间在这块重点训练。

一切的基础，带你迈入编程的大门。

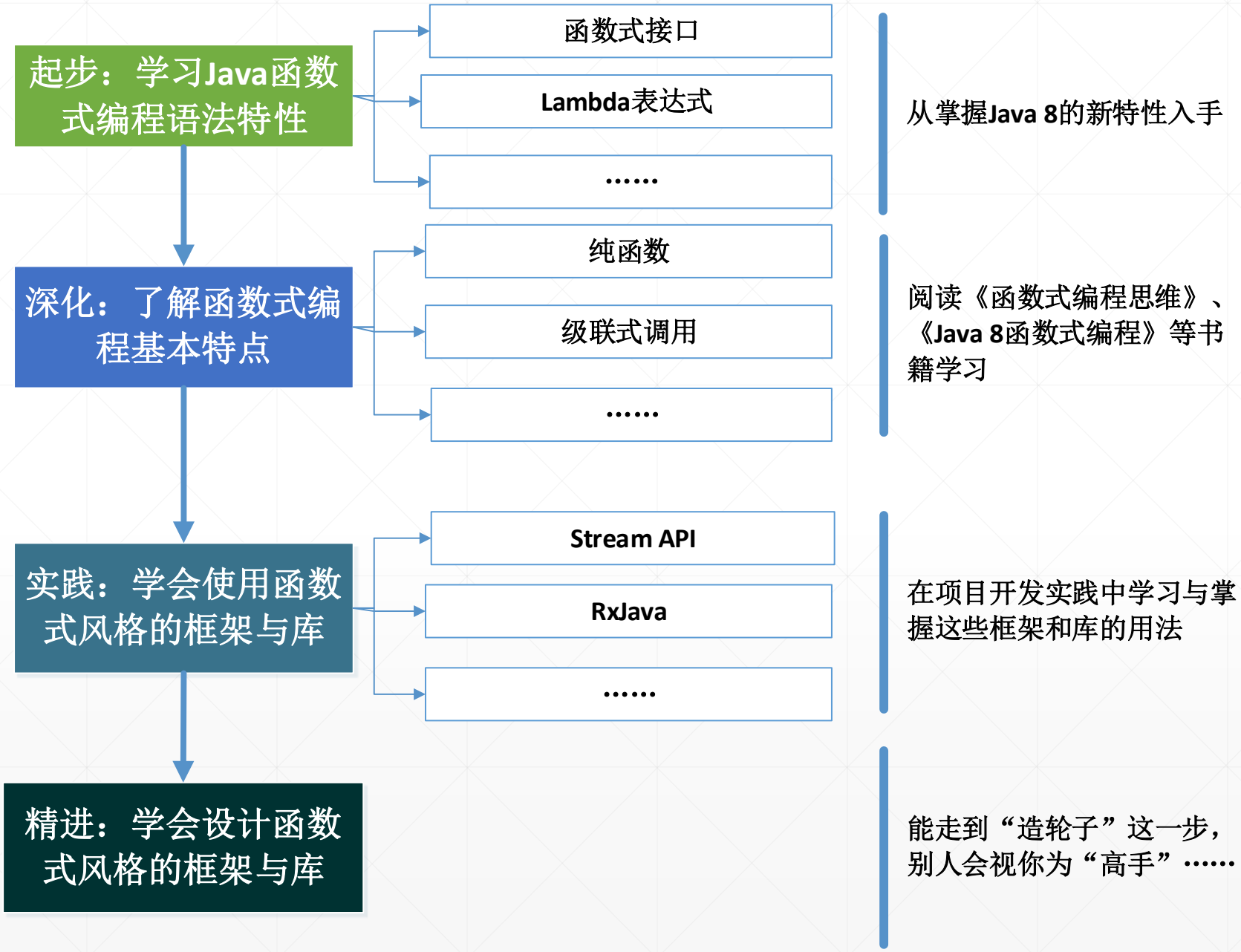
结构化编程技能的刻意训练计划



面向对象开发技能的刻意训练计划



函数式编程技能的刻意训练计划



Java学习全局路线图

初学者

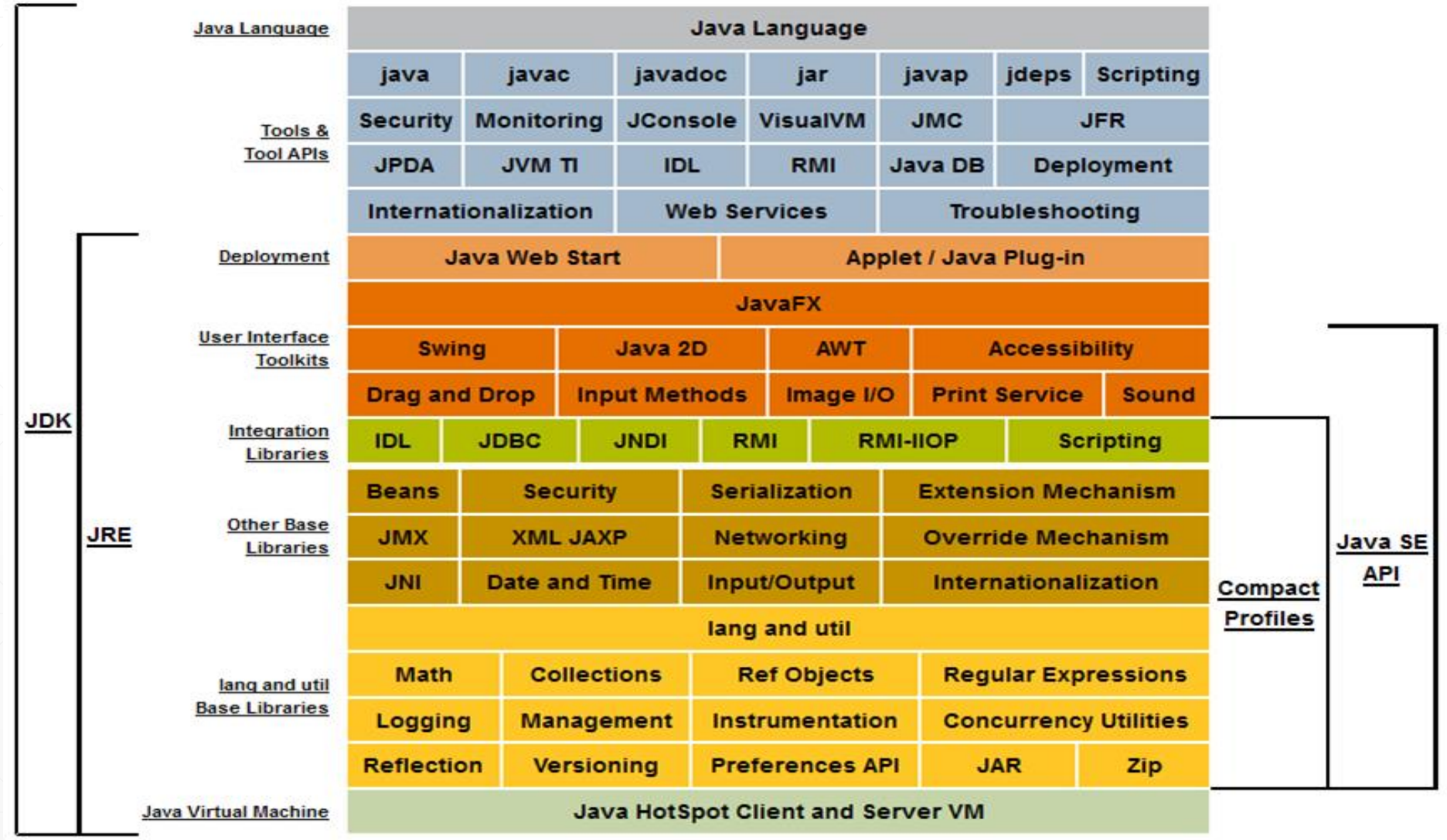
Java SE → android → Java EE → web前端 → 开发移动互联应用 → 开发智能应用

走完这条完整的学习路线，可以构建出一个比较完备的知识框架，培养出当前时代软件开发所需的职业技能。

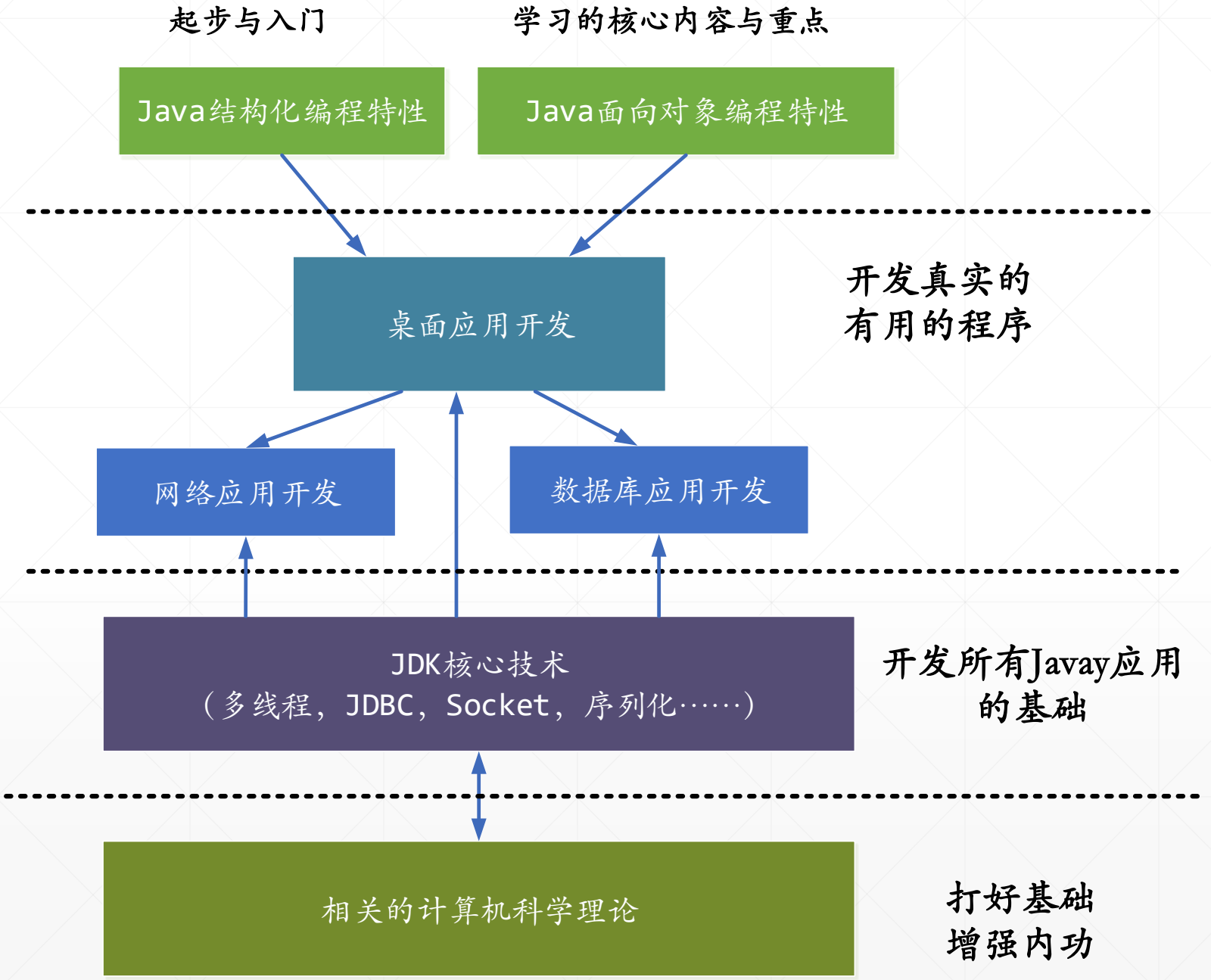
合格的Java程序员

迎接下一波技术革命浪潮

Java SE技术平台的主要组成部分（学习内容）



Java SE知识关联图（初级阶段）

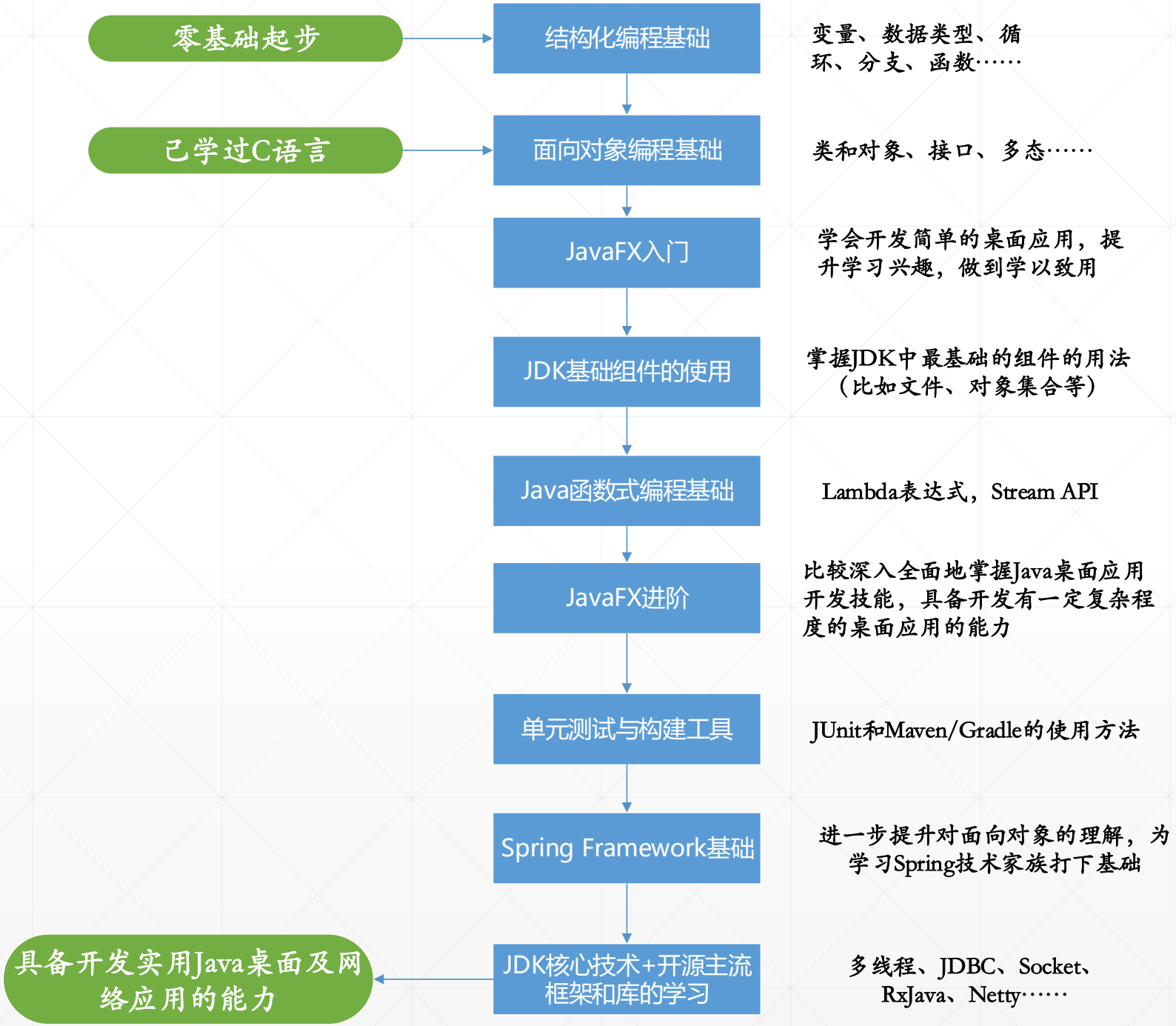


图例说明

A → B

A为B提供支持

JavaSE学习路线图（初级阶段）



2 Java程序员的“修仙”之路

Java SE自学指南

本次Live将按照以下顺序分块介绍每部分的学习要点，并给出自学建议：

1. 零基础结构化编程起步
2. 面向对象编程技能训练
3. 函数式编程技能训练
4. 桌面应用开发技术
5. 数据库开发技术
6. 网络应用开发技术

2.1 零基础编程起步

零Java编程基础的起步

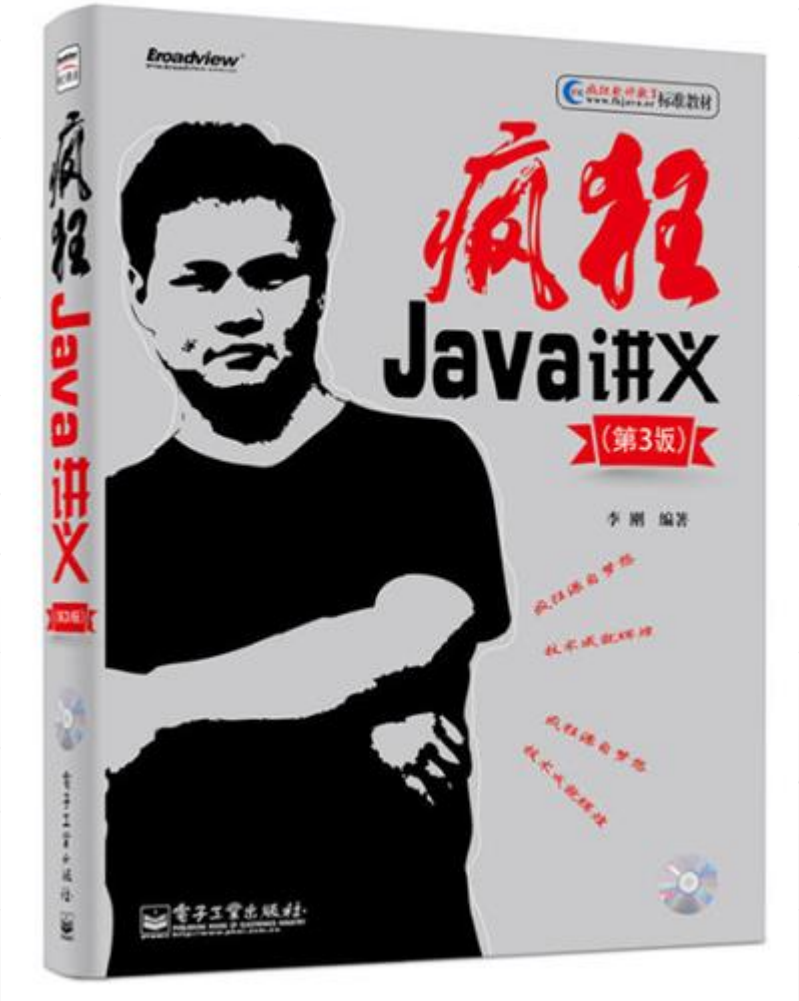
1 弄清关键的术语与概念

2 开始折腾计算机

- 下载和安装相关软件，搭建Java学习与开发环境
- 学会使用命令行开发Java程序
- 掌握IDE的基本用法，走一个HelloWorld流程

3 了解编程这件事，构建第一印象

- 数据类型、变量的内存模型、运算符与表达式
- 程序控制结构
- 输入数据与输出结果的方法
- Java程序结构：类、源代码文件与程序入口点


Broadview
疯狂
Java讲义
（第3版）
李刚 编著
疯狂Java讲义
电子工业出版社

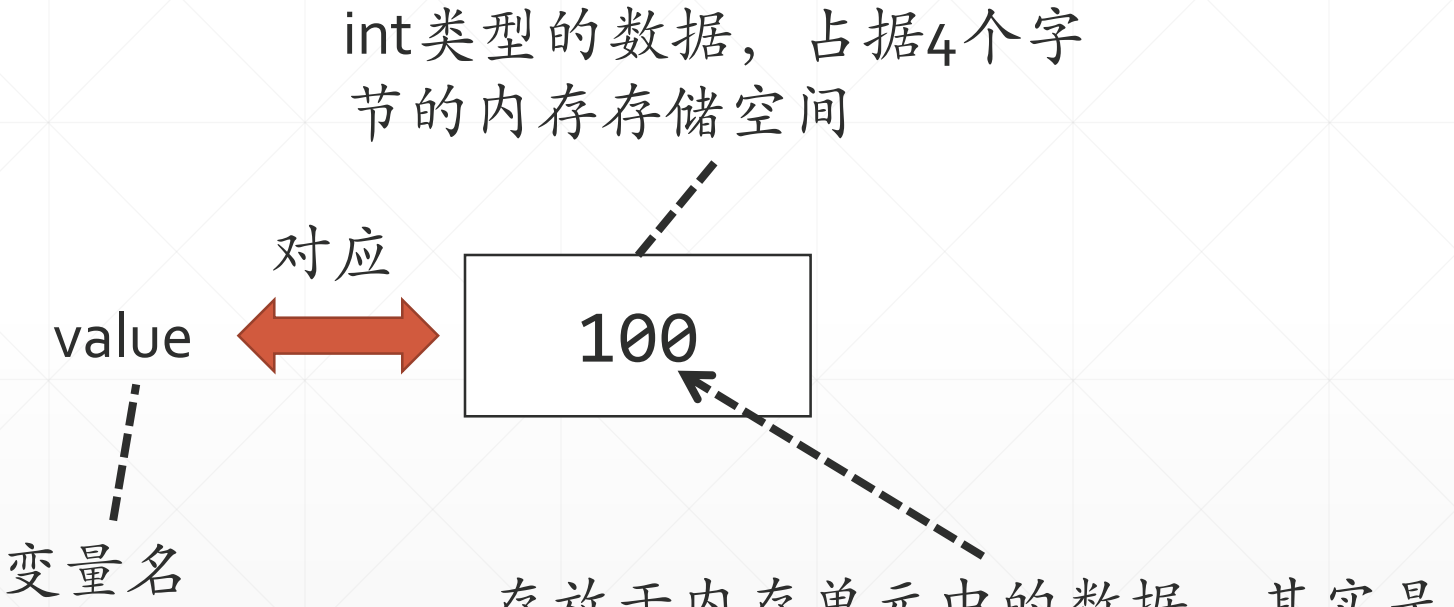
初学入门


Java核心技术 卷I
基础知识（原书第10版）
Core Java Volume I–Fundamentals
[美] 凯S. 霍斯特曼（Cay S. Horstmann）著
Java核心技术 卷II
高级特性（原书第10版）
Core Java Volume II—Advanced Features
[美] 凯S. 霍斯特曼（Cay S. Horstmann）著
Pearson
机械工业出版社
China Machine Press

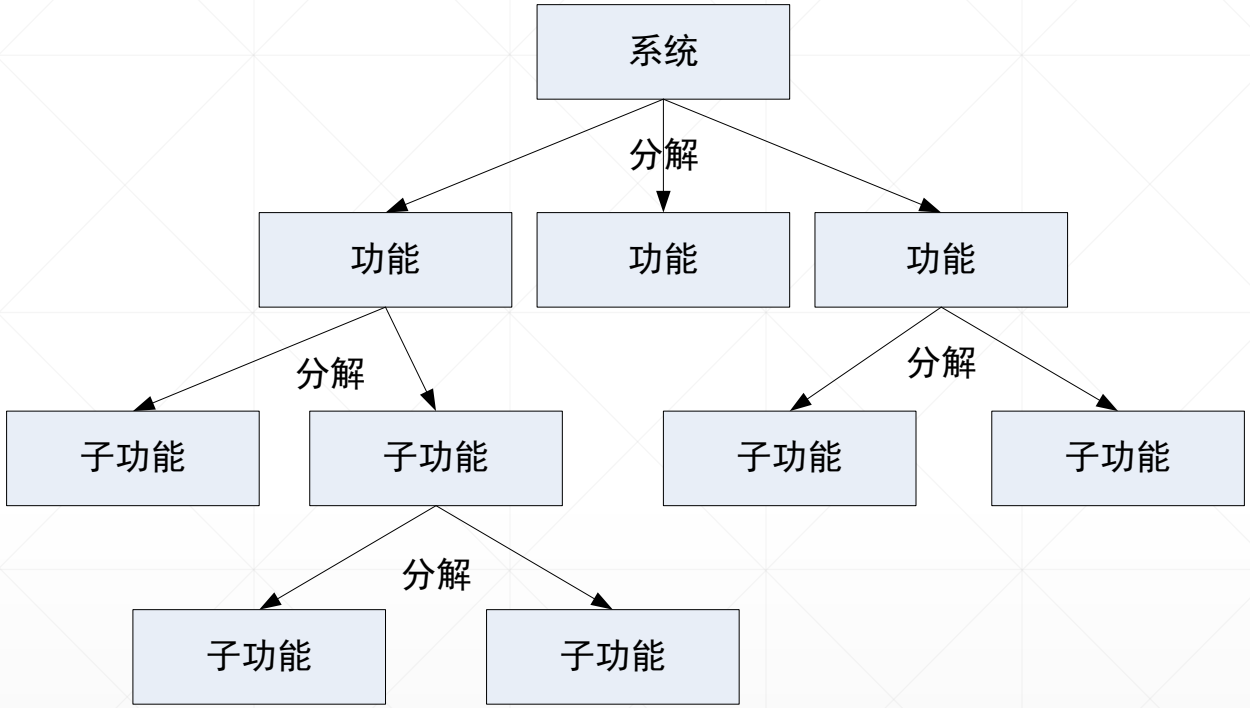
开发参考

零基础编程起步-变量的内存模型

```
int value = 100;
```



结构化编程的功能分解法



细化分解到每个子功能都可以使用一个函数（或若干个函数配合）实现。

结构化开发的具体实施

调用已有的函数模块

开发自己的函数模块

将所有模块装配起来

递归编程技能训练示例

1．使用递归计算整数n的阶乘：

n! = 1*2*3*…*n = (n-1)!*n

调用上面的成果，计算出组合数：

$$C_n^k = \frac{n!}{k!(n-k)!}$$

2。 直接使用以下公式用递归计算：

$$C_{n+1}^k = C_n^{k-1} + C_n^k$$

3. 在学习完Java文件操作相关的功能之后，编写一个程序，用户给定一个文件夹，能递归地列出其中所有的子文件夹和文件。
在这个递归的过程中，还能完成一些额外的数据处理工作，比如统计每个子文件夹的容量，找出最大的文件等等。

递归是Java程序员必备技能，必须多练，达到能随手写出这几道递归编程练习题的程度。

2.2 面向对象编程技能训练

Java面向对象特性学习要点-1

深刻理解类和对象的概念

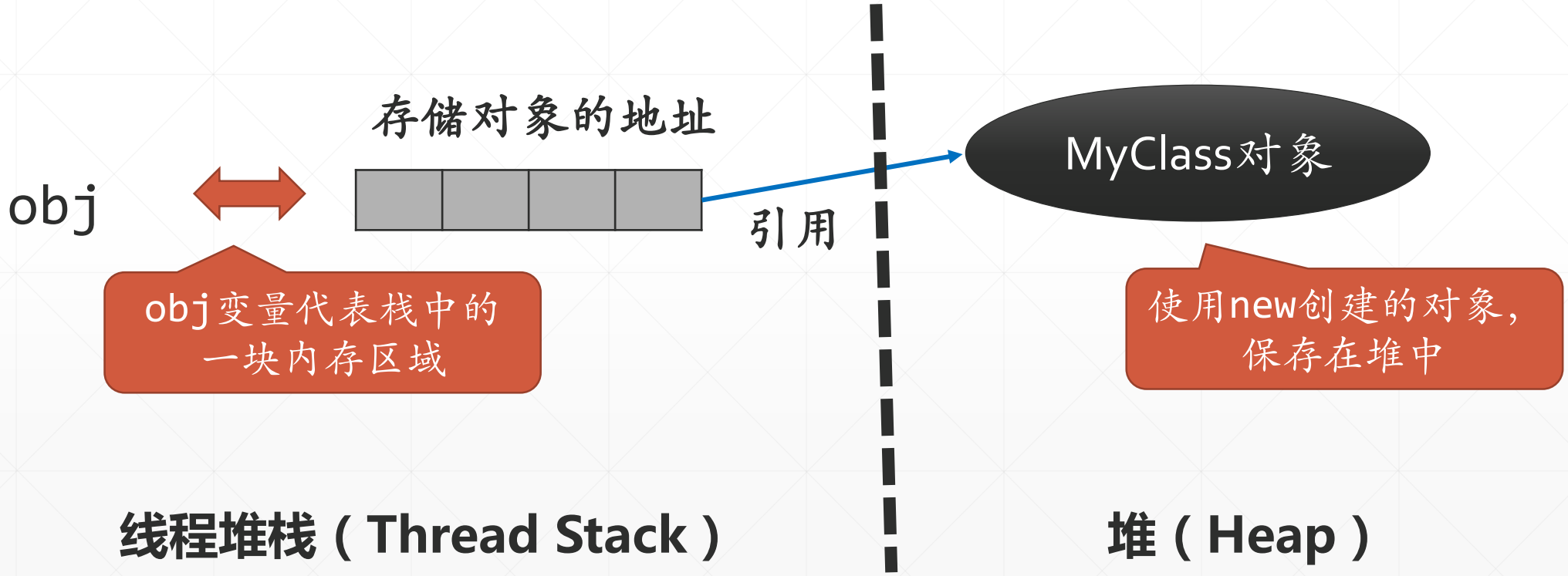
系统掌握面向对象基本特性与Java编程技巧

- **抽象**：知道如何将现实事物的特性抽象为类
- **封装**：学会编写Java类，理解信息隐藏原则
- **继承**：掌握父类子类之间的关系与混合使用
- **多态**：理解抽象基类与接口，活用于编程中

深刻地理解对象变量与对象

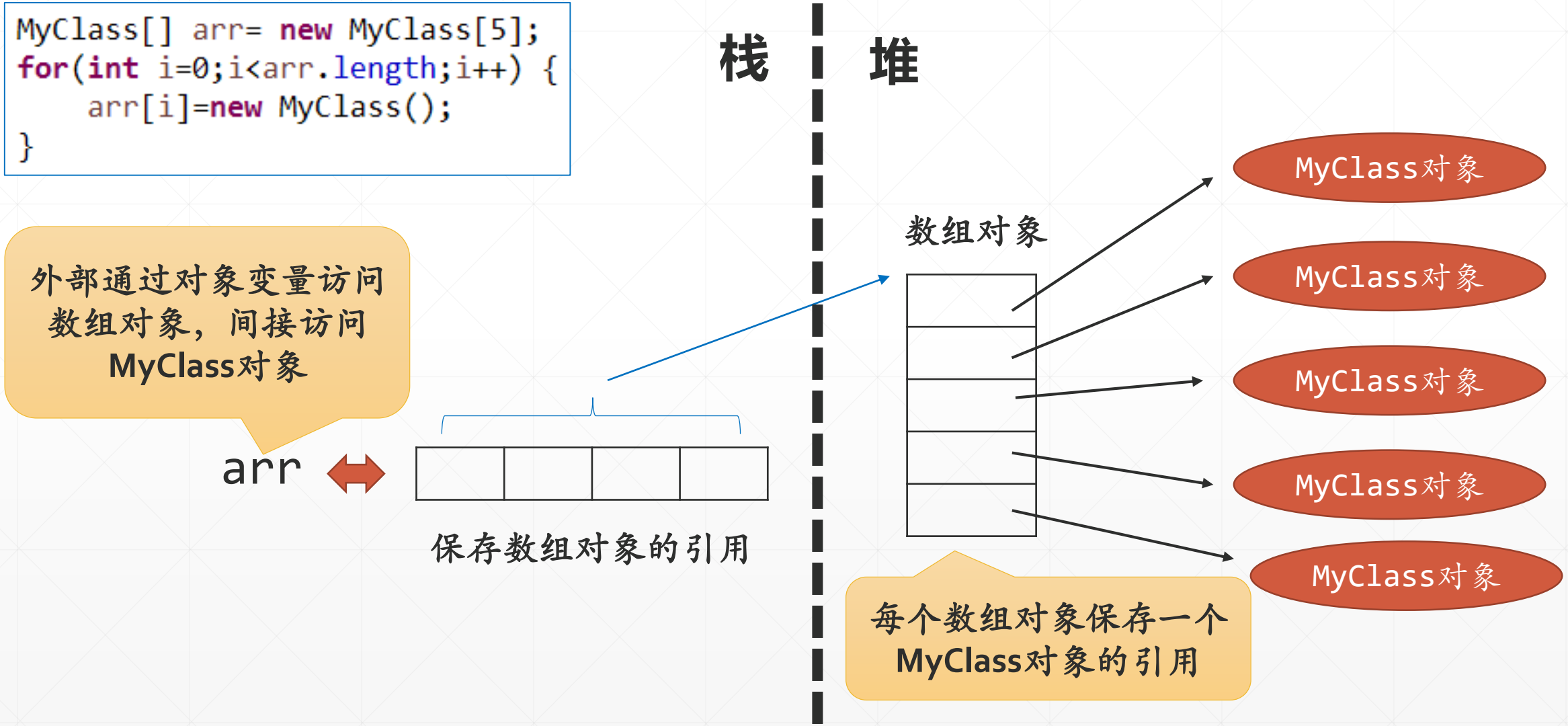
对象变量的内存模型

```
MyClass obj = new MyClass();
```

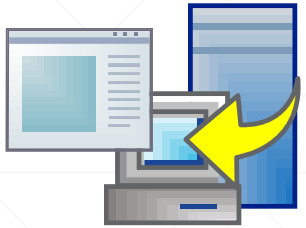


对象数组的内存模型

```
MyClass[] arr= new MyClass[5];
for(int i=0;i<arr.length;i++) {
    arr[i]=new MyClass();
}
```



面向对象“抽象”与“多态”特性应用的实例



基于Netty开发
的网络应用程序

Netty提供的统一公用API
（Channel、ChannelHandler、ChannelPipeline）

阻塞式网络I/O操作

非阻塞式网络I/O操作

Java面向对象特性学习要点-2

对象组合

- 两种对象组合方式
- 自引用类
- 对象注入与IoC容器

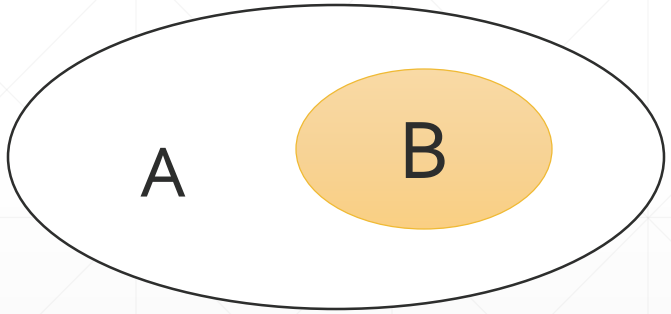
对象集合

对象之间的协作与消息交换

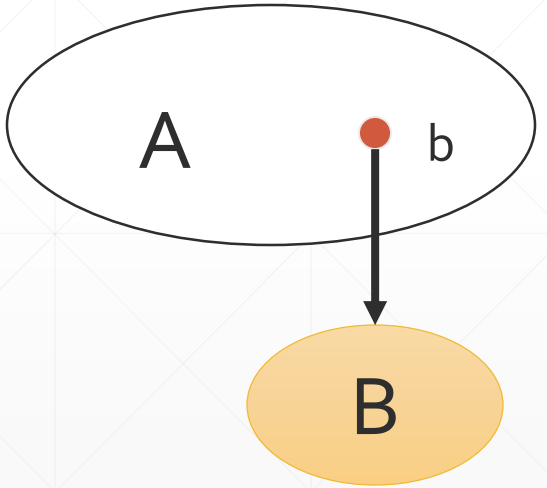
对象与组件化开发

对象组合

- 一个对象包容或引用另一个对象，称为“**对象组合**”。
- 有两种典型的对象组合方式：



方式一：A对象完全包容B对象，容器对象A管理被包容对象B的生命周期



方式二：B对象是独立的，A对象引用现有的B对象

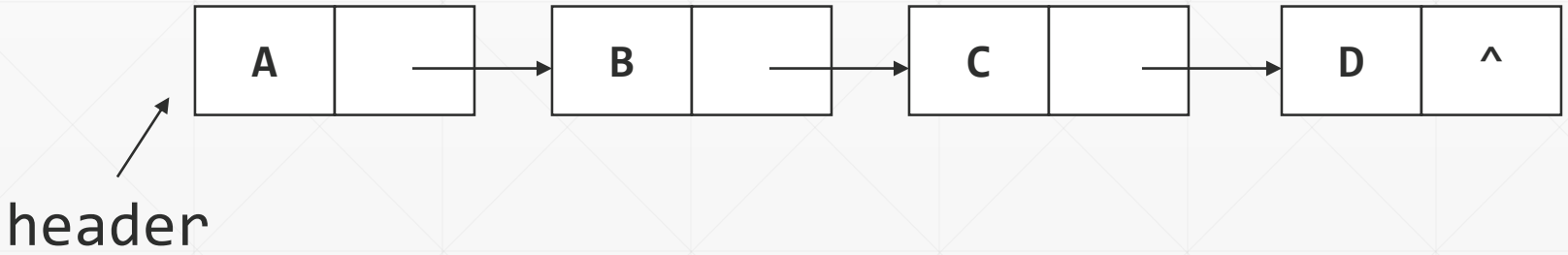
对象组合的特例——自引用类

```
public class LinkNode {
    public LinkNode next;
}
```

自引用类

Self-Reference Class

自引用类经常用于实现诸如链表，
树、图等数据结构



对象组合引申出的新概念——依赖注入

```java
//此接口抽象出了程序中需要用到的某些功能
interface SomeService{
    //...
}
//MyService类具体实现了此接口
class MyService implements SomeService{
    //...
}
//MyClass需要someService接口所定义的功能
class MyClass{
    private SomeService service;
    //MyClass自己不new一个MyService对象，而是要求
    //外界“准备”好一个实现了SomeService的对象，
    //通过构造方法“注入”进来
    public MyClass(SomeService service) {
        this.service=service;
    }
}
```

只要你按照左图所示的模式编写代码，Spring Framework就能自动地帮助你创建并装配对象！

JDK中对象集合的主要类型

1. **Set**：无序、不可重复的集合
2. **List**：有序、可重复的集合
3. **Map**：具有映射关系的集合
4. **Queue**：具有队列特性的集合

上述集合又可以分为“**线程安全的**”和“**不是线程安全的**”两大“阵营”。

对象集合的学习要点

（1） 掌握对象集合的基本操作

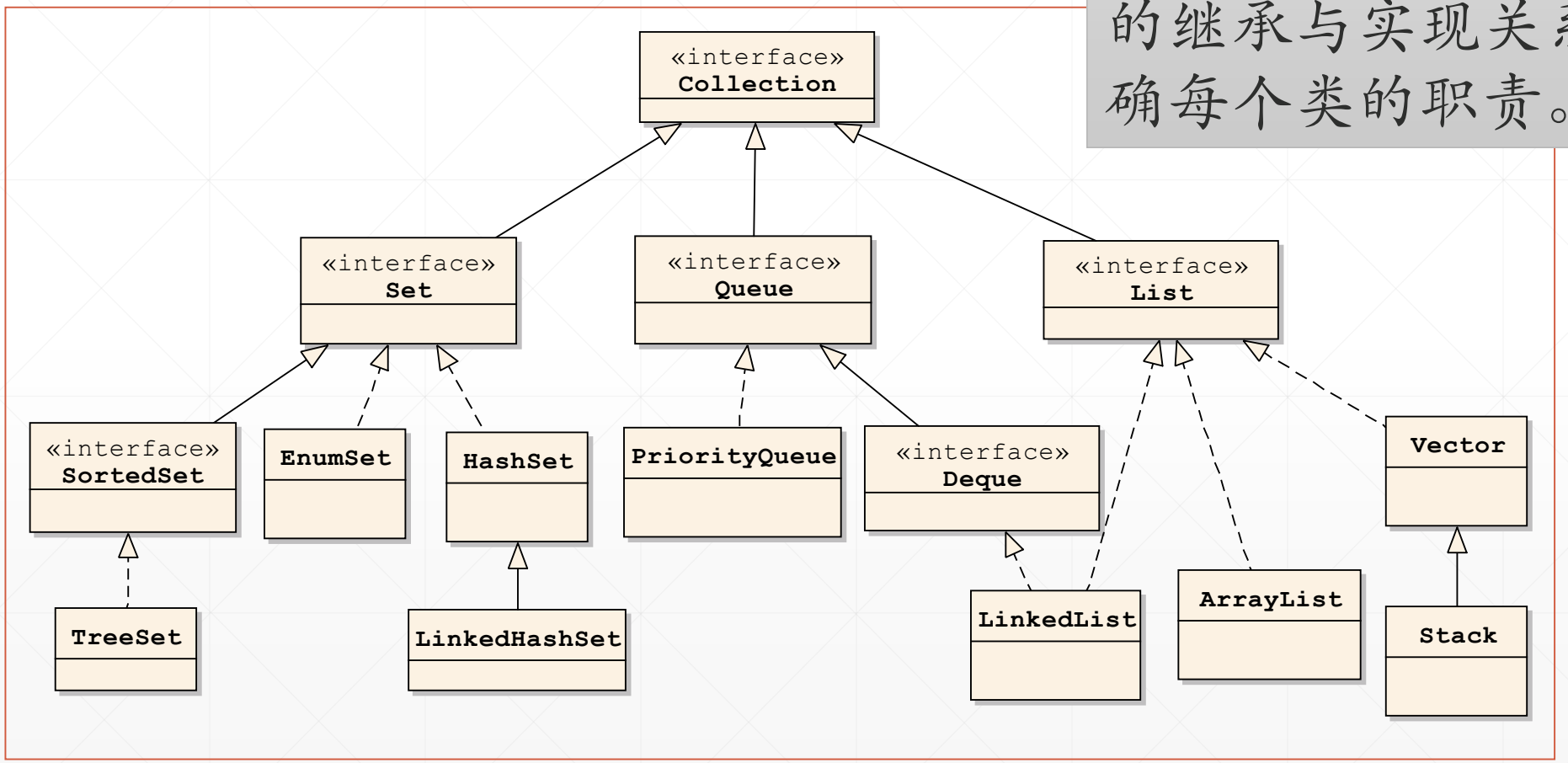
- 增、删、遍历、查找、排序.....
- 要求放入集合中的对象，必须支持**判等**和**大小比较**。

（2） 弄清楚JDK中与集合相关的接口与类的继承和实现关系

（3） 明确JDK中各种集合的应用场景和优缺点以便选用

JDK中集合相关的部分类型

弄清楚各个类和接口之间的继承与实现关系，并明确每个类的职责。



每个集合填一张表……

项目	说明
集合名称	HashSet
特性	将对象的哈希码作为访问对象的“索引（index）”
应用场景	需要很快的查找和增删操作
实现的接口	Set、Cloneable、Serializable
重要方法	add、remove、contains、size、iterator
内部实现	内部包容一个HashMap

从JDK中选一些重要的集合，逐项填写上面这张表，然后再有针对性地写一点小的Demo，这个集合的用法就搞掂了。

对象之间的协作与消息交换

对象之间的三种消息交换方式：

一对一	又可以再进一步地细分为三种情况：A发信息给B、A委托B收集一些信息、A与B双向（实时）交换信息
一对多	一个对象发消息给多个对象，或者是多个对象发消息给一个对象
多对多	多个对象之间进行“群聊”

编程实现对象之间一对一消息交换的两种基本方法

直接法

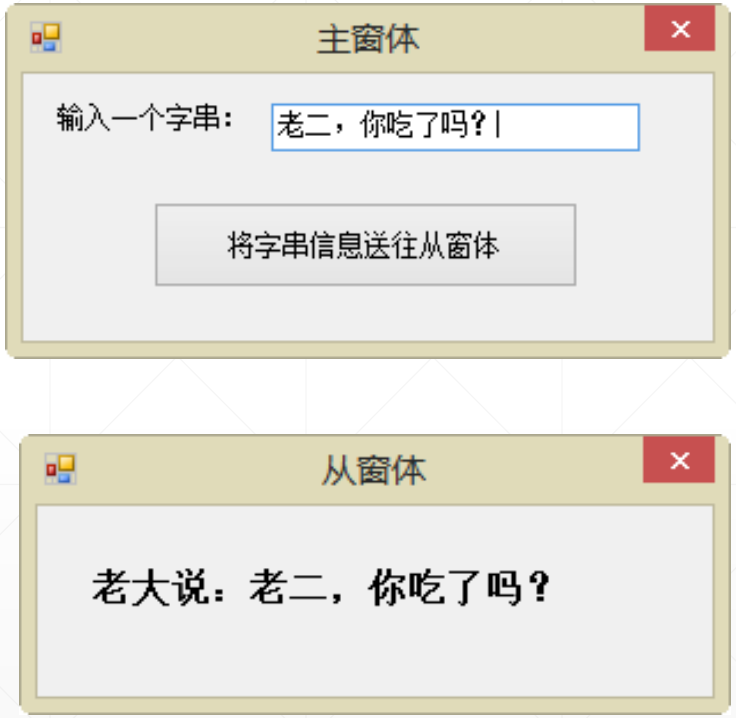
① 对象A执有对象B的引用

② A通过对象B的引用存取对象B的公有属性，或者调用对象B的公有方法

回调法

B事先准备好一个用于接收消息的方法，然后将这个方法以某种方式告诉A，然后A在消息准备好之后“**回调（call back）**”B告诉自己的那个方法。

对象间一对一信息传送——单向传送



“消息框”示例

“对话框”示例

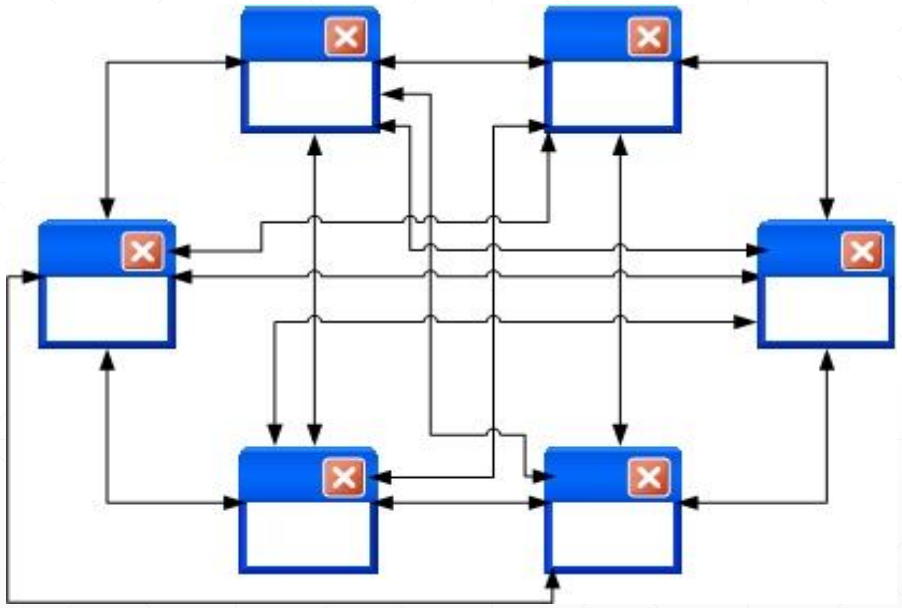
两个对象间信息的双向实时传送示例

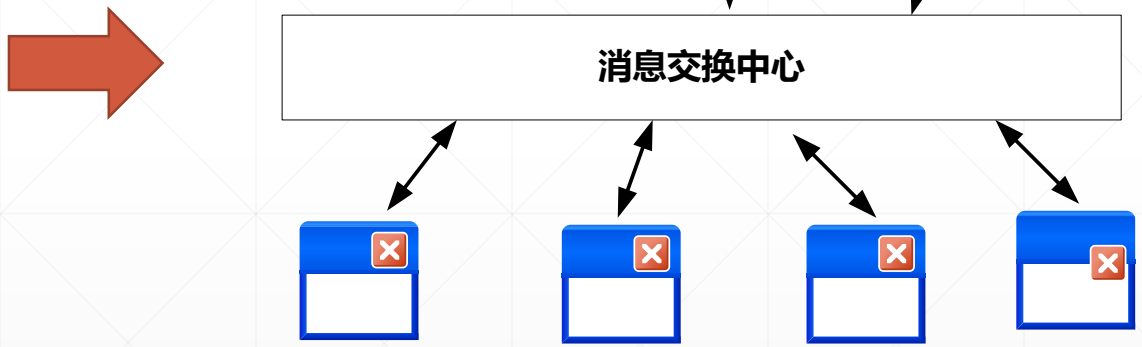
示例的特点：

两个窗体之间实现了即时的双向互动和同步，在任何情况下，主从窗体所显示的百分比值都同步显示。

一对多的消息传送——“广播”

“多对多”的对象信息交换带来的挑战





如果有多个对象之间相互交换信息，很容易形成一个复杂的通信网，难以实现与维护。

通过“断开”两个对象之间的直接关联，引入中间媒介，可以很好地实现对象之间的“多对多”通讯。

刻意练习：尝试着自己实现面向对象的常用编程套路

对象池

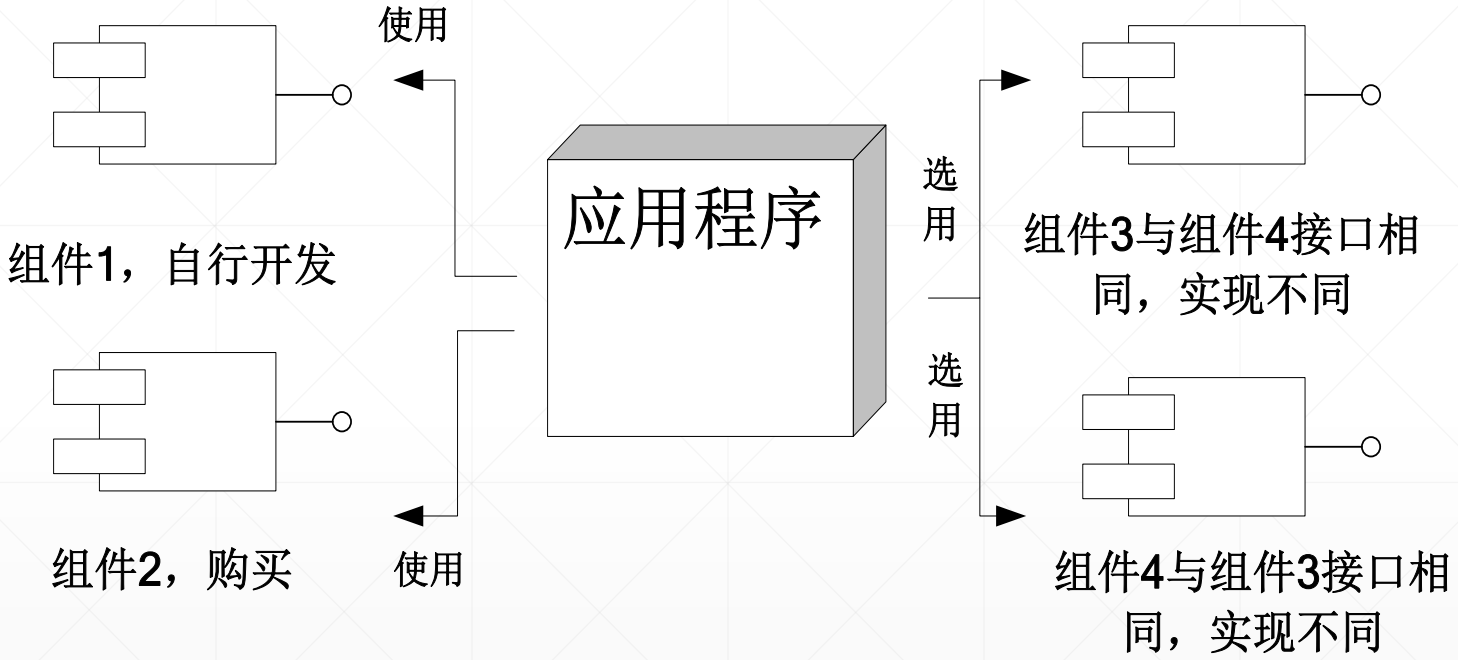
数据缓存对象

管道

事件队列与消息循环

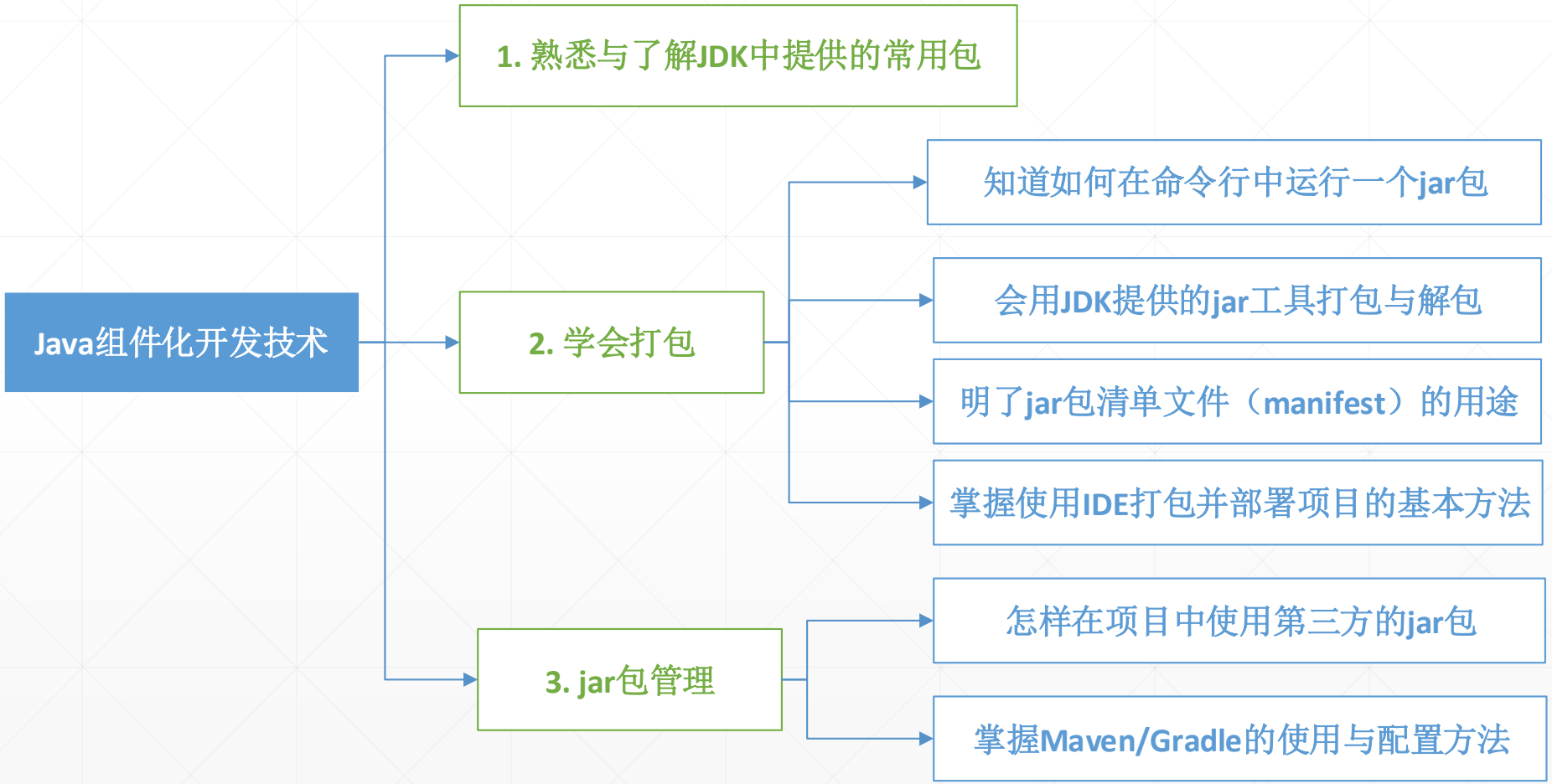
请在各种开发框架中寻找上述编程套路出现的地方，并且尝试着自己实现这种编程套路，完成这些任务，可以有效地提升你的面向对象软件开发技能。

组件化软件的开发方式

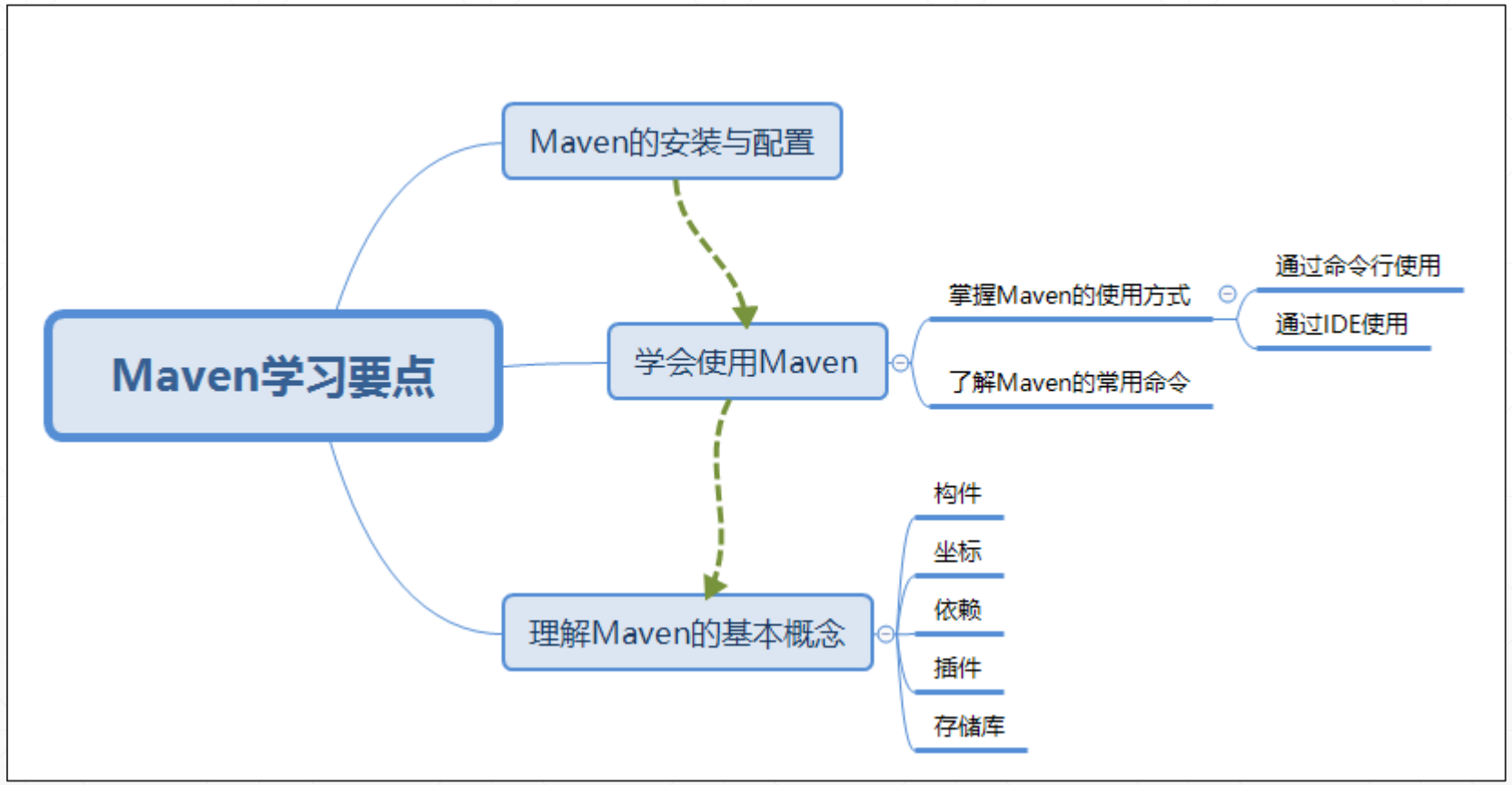


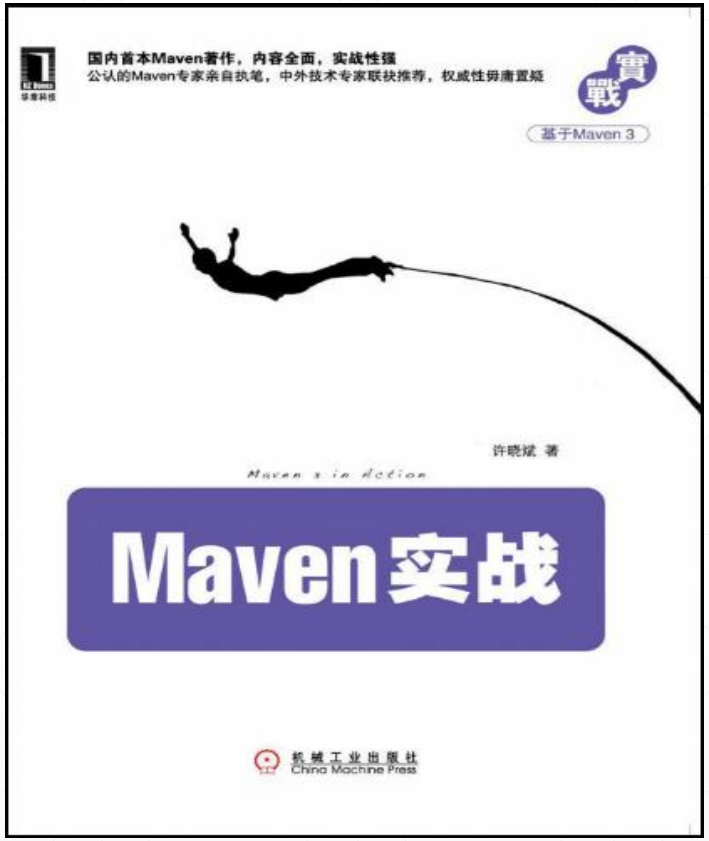
现代软件系统，大多基于各种组件，以搭积木的方式构建出来。

Java组件化技术知识网络与学习路线



Maven的学习

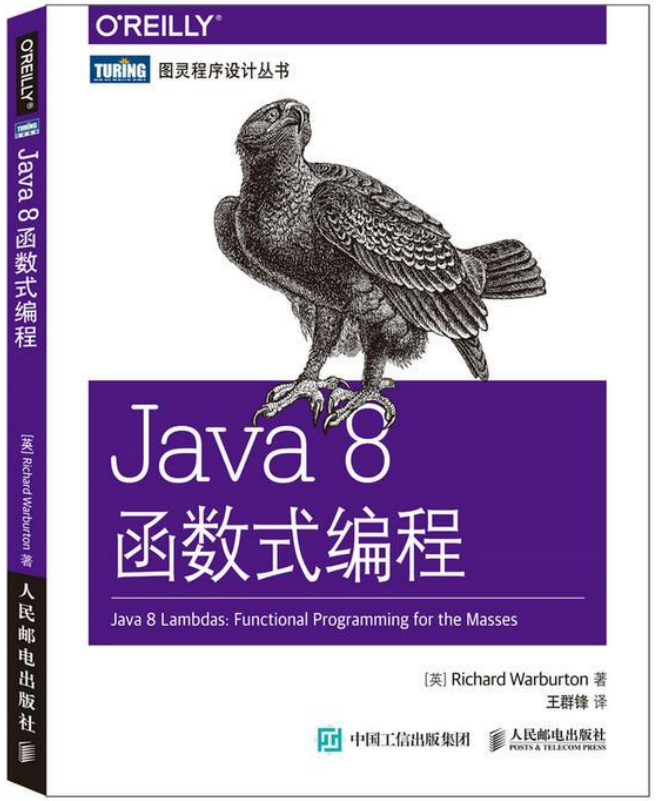




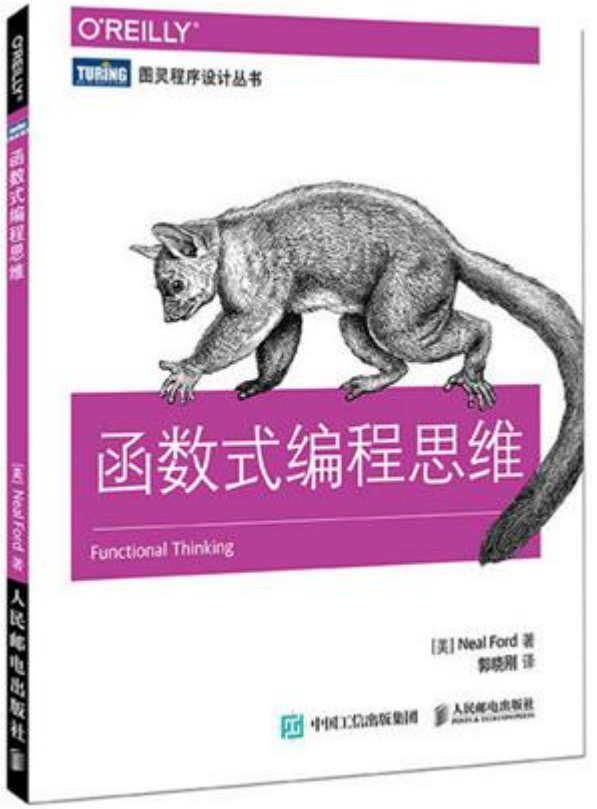
注：上图中的绿色箭头表示学习顺序

2.3 函数式编程技能训练

Java函数式编程技术要点与学习路线


O'REILLY
TURING 图灵程序设计丛书
Java 8 函数式编程
[英] Richard Warburton 著
人民邮电出版社
Java 8
函数式编程
Java 8 Lambdas: Functional Programming for the Masses
[英] Richard Warburton 著
王群锋 译
中国工信出版集团
人民邮电出版社

入门


O'REILLY
TURING 图灵程序设计丛书
函数式编程思维
[美] Neal Ford 著
人民邮电出版社
函数式编程思维
Functional Thinking
[美] Neal Ford 著
郭晓刚 译
中国工信出版集团
人民邮电出版社

进阶与深入

Stream API

```
// 使用Stream API的并行编程方式
static void filterWithParallelStreamAPI() {
    System.out.println("线程："+Thread.currentThread().getId()
            +"启动处理工作");
    long count = Student
            .getStudents()
            .parallelStream()
            .peek(stu -> {
                System.out.println("线程" + Thread.currentThread().getId()
                        + "正在处理" + stu);
            }).filter(stu -> stu.getFrom().equals("北京")).count();
    System.out.println("线程："+Thread.currentThread().getId()+
            "显示处理结果：来自于北京有学生有" + count + "名");
}
```

Stream API具有函数式编程风格。

Stream API中的Stream，与IO操作中的Stream，是两回事！

- Stream API主要用于操作对象集合。
- Stream API提供了一系列的支持级联调用方法，可以用很少的代码完成复杂的数据处理功能。
- Stream API支持并行处理。

Java程序员必备之函数式编程技能清单

1. 熟练使用Lambda表达式编程
2. 能写出没有副作用的函数（即纯函数），设计出只读类
3. 能写出支持链式（级联）调用的高阶函数（返回函数的函数）
4. 熟练使用Stream API操控对象集合，完成各种数据处理工作
5. 能熟练使用RxJava等具有函数编程风格的库或框架编写程序

2.4 Java桌面应用开发技术

JavaFX学习中的关键技术点

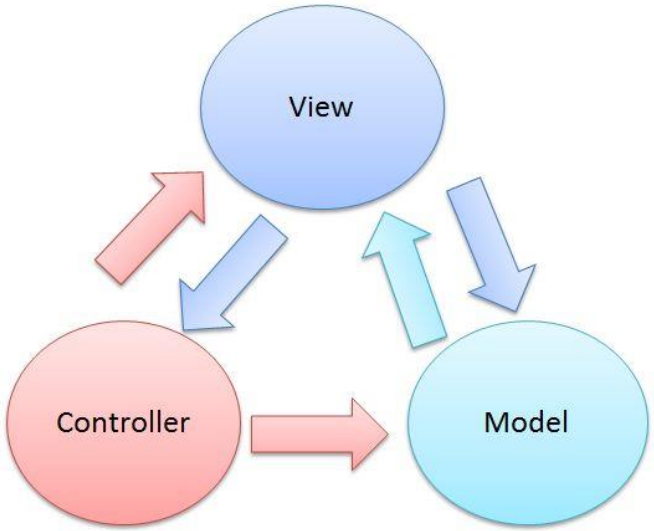
基于MVC模式的JavaFX应用程序架构

数据绑定技术

JavaFX线程模型

图形与多媒体

JavaFX中的MVC设计模式



视图由一个FXML文件来定义，每个视图关联着一个**控制器**，在控制器中可以引用在视图中的控件，并且为特定的控件编写事件响应代码。

刻意训练方法：
编写Java多窗体应用程序，在这些窗体之间实现一对一、一对多的信息传送。

JavaFX数据绑定“编程模式”

绑定数据源
（具备改变通知能力的
数据对象与数据集合）

JavaFX数据绑定机制

JavaFX应用程序
的UI界面

ObservableList<T>、
ObservableValue、
JavaFX Bean Property

Binding对象

包容TableView、Label等支持数据绑定的控件

只要写一个程序，就能搞掂JavaFX数据绑定技术的学习

示例参考界面

编写一个经典的桌面版CRUD（增删改查）数据库应用程序，就能搞掂JavaFX数据绑定机制，还买一赠一，顺便把JDBC访问数据库的技术也一并学了练了。

JavaFX多线程应用的编程原则与典型套路

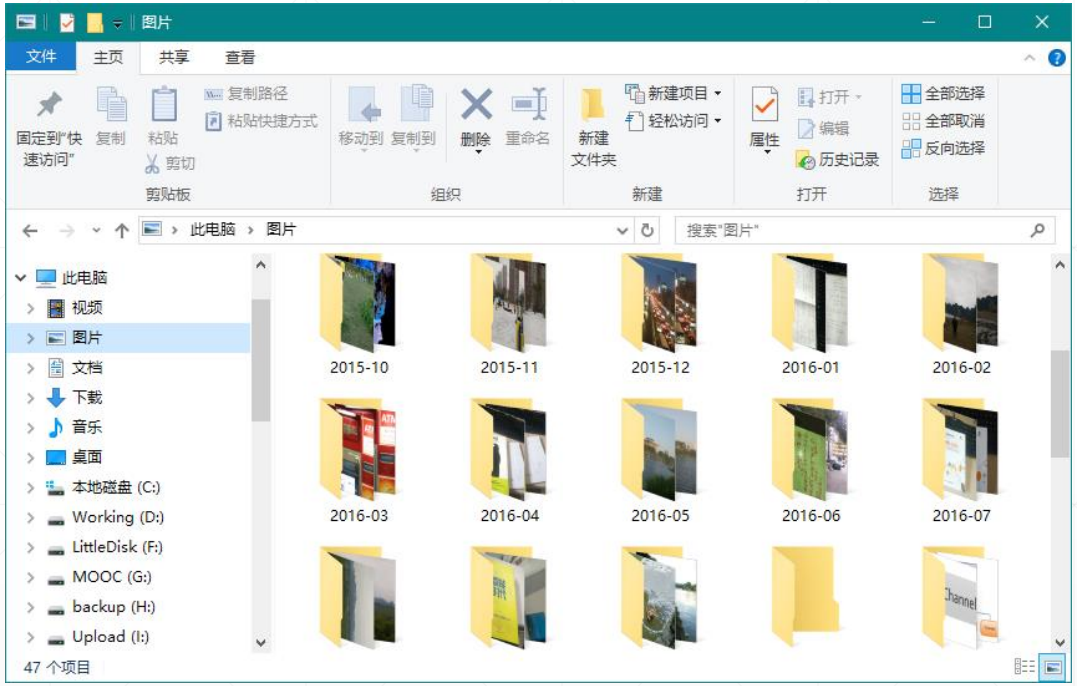
不要在UI线程之外访问UI控件！

编程套路

在JavaFX中，可以在其他线程中new一个node，附加到Scene，之后在JavaFX Application Thread访问它。

可以使用后台工作线程提取相关的数据，将其加入到数据集合中，然后利用JavaFX数据绑定机制的特性刷新UI界面。

一个程序搞掂JavaFX多线程开发技术的学习

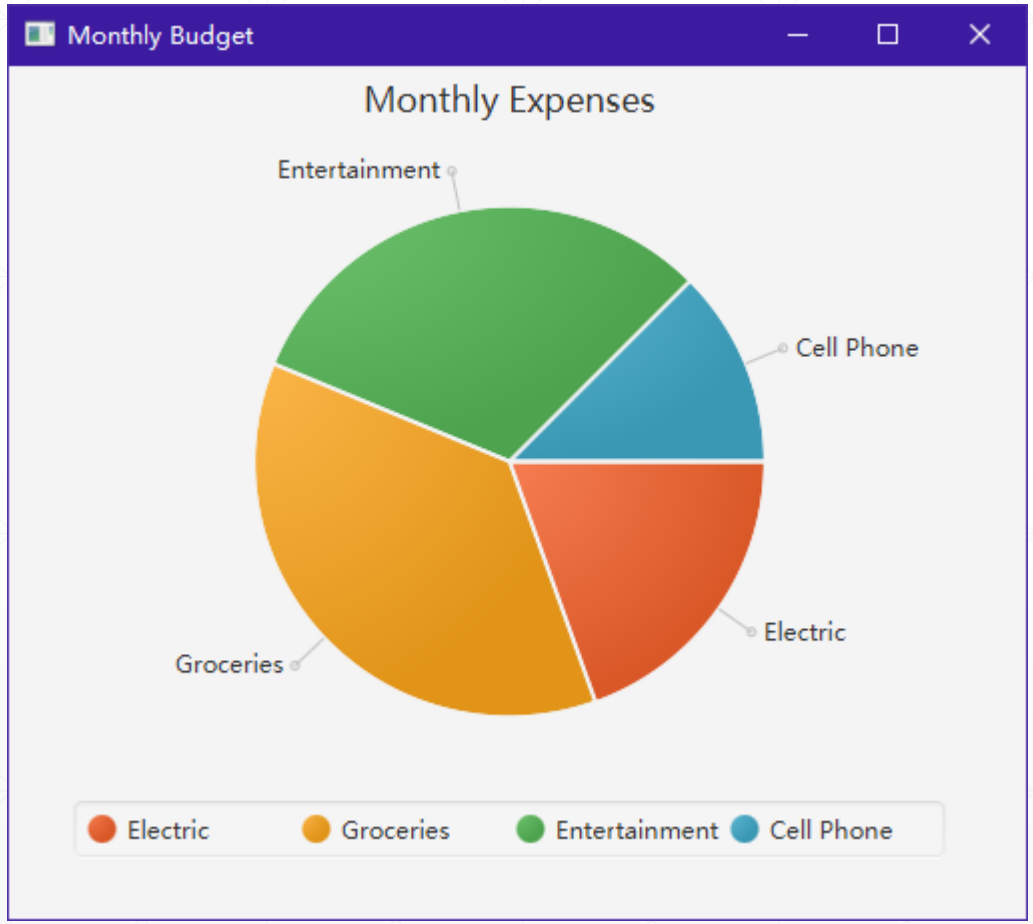


模仿Windows文件资源管理器，用JavaFX自己写一个，就可以弄熟N多技术内容：

- UI控件的使用与界面布局技巧
- 多线程（异步）后台访问文件系统
- 利用数据绑定机制动态更新UI界面，
- ……

类似于这样的程序，写一个就足够让你的编程能力提升一个台阶了。

JavaFX的图形与多媒体



JavaFX提供了2D绘图支持，也拥有动画处理能力，可以方便地开发各种数据可视化程序或小游戏。

JavaFX的学习资源

JavaFX的学习资源主要来自官网：

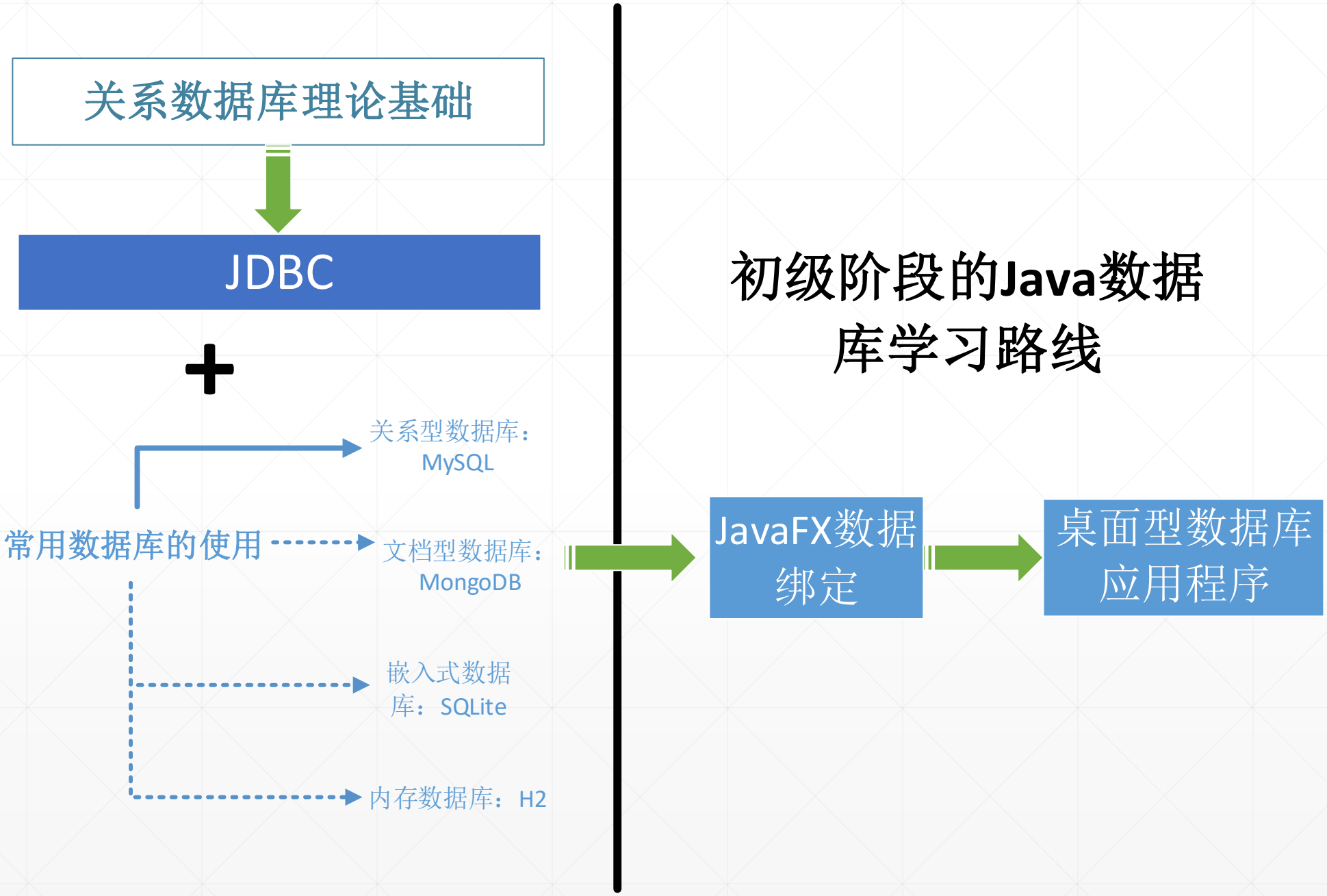
http://docs.oracle.com/javase/8/index.html

国外出版了一些JavaFX技术书籍，但国内没有引进。

使用Google、百度等搜索引擎可以找到一些零散的学习资源。

这是训练你搜索学习资源，提升自学能力的一个机会！

2.5 数据存取技术


关系数据库理论基础
JDBC
+
常用数据库的使用
关系型数据库：MySQL
文档型数据库：MongoDB
嵌入式数据库：SQLite
内存数据库：H2
初级阶段的Java数据库学习路线
JavaFX数据绑定
桌面型数据库应用程序

JDBC核心类型

核心类型	说明
java.sql.DriveManager	负责装载/卸载驱动程序
java.sql.Connection	与数据库建立连接
java.sql.Statement	在一个给定的连接中执行SQL语句
java.sql.PreparedStatement	用于执行预编译的SQL命令
java.sql.CallableStatement	用于调用数据库中存储过程
java.sql.ResultSet	保存SQL命令的执行结果

JDBC访问数据库的基本步骤

加载JDBC驱动程序

创建数据库连接

执行SQL语句

接收并处理SQL的返回结果

关闭创建的各个对象

JDBC编程模式

典型的JDBC访问数据库代码：

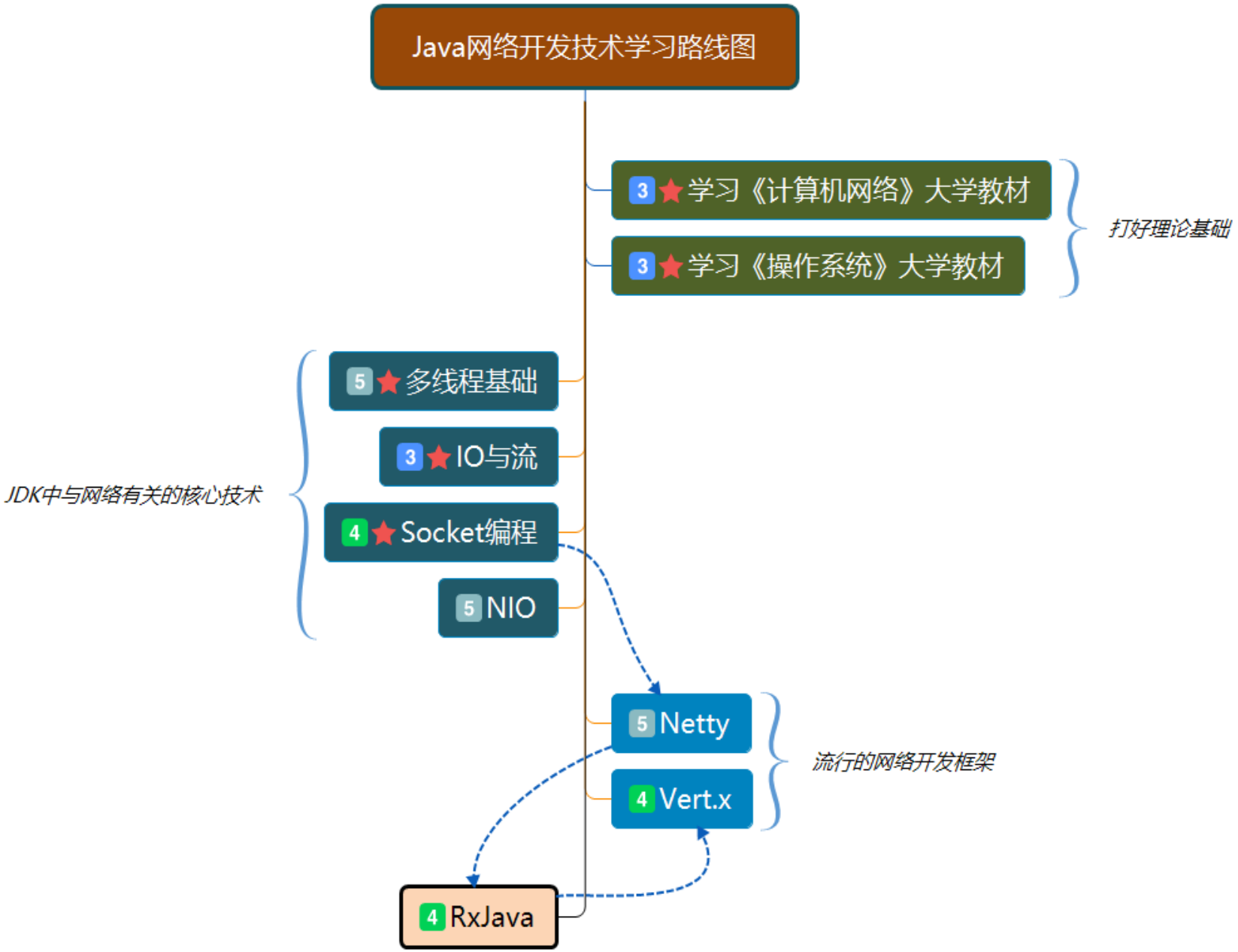
```
try{

    //获取数据库连接
    //启动一个事务（transaction）
    //创建和执行SQL命令
    //处理提取到的数据
    //提交（commit）事务

}catch(SQLExpression e){

  //当有异常发生时，回滚（rollback）事务

}finally{

 //关闭各种数据库资源对象
 //比如connection,statment等

}
```

推荐的编程模式：

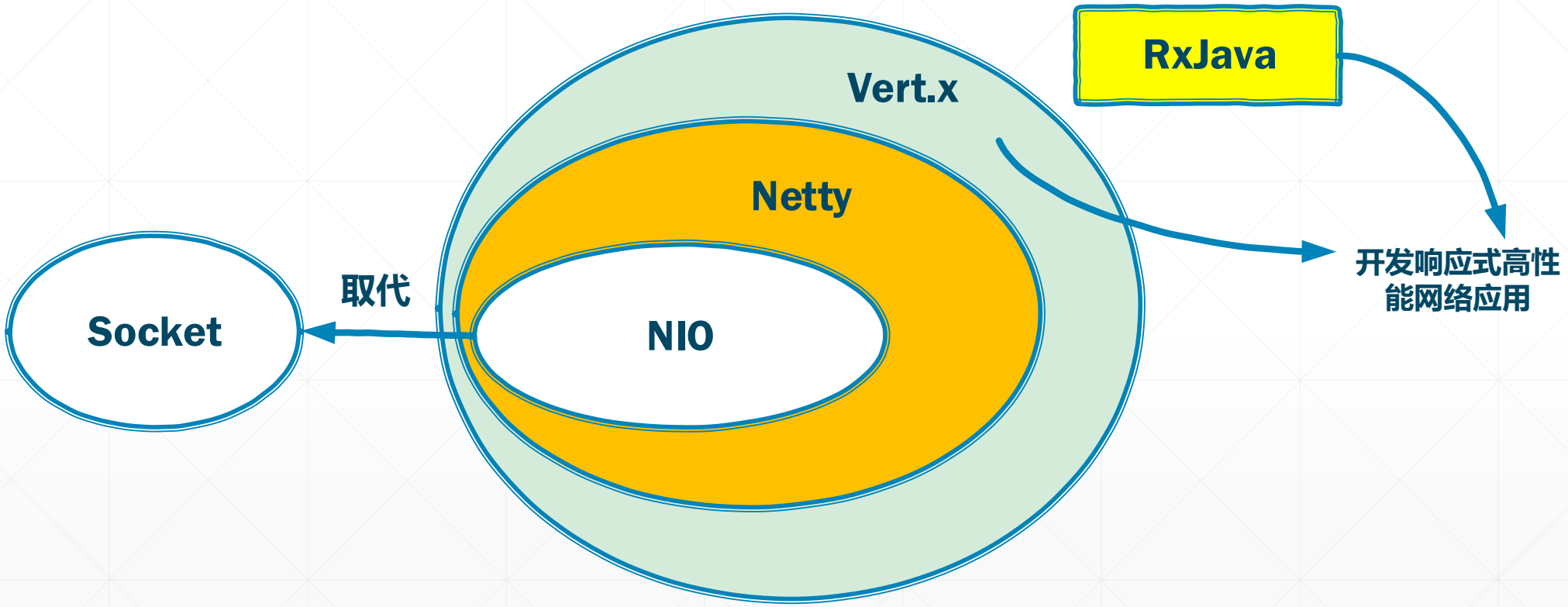
```
//不要Return一个ResultSet
//应该返回一个对象集合供上层组件使用
List<MyDataClass> getData(){

      List<MyDataClass> list= new ....;
      //访问数据库，提取数据放入ResultSet
      //遍历ResultSet,
      //每行创建一个MyDataClass对象
      //将MyDataClass对象加入list中
      return list;

}
```

实际开发中，常用套路是使用Repository设计模式来封装数据存取代码，请利用互联网自学这块内容。

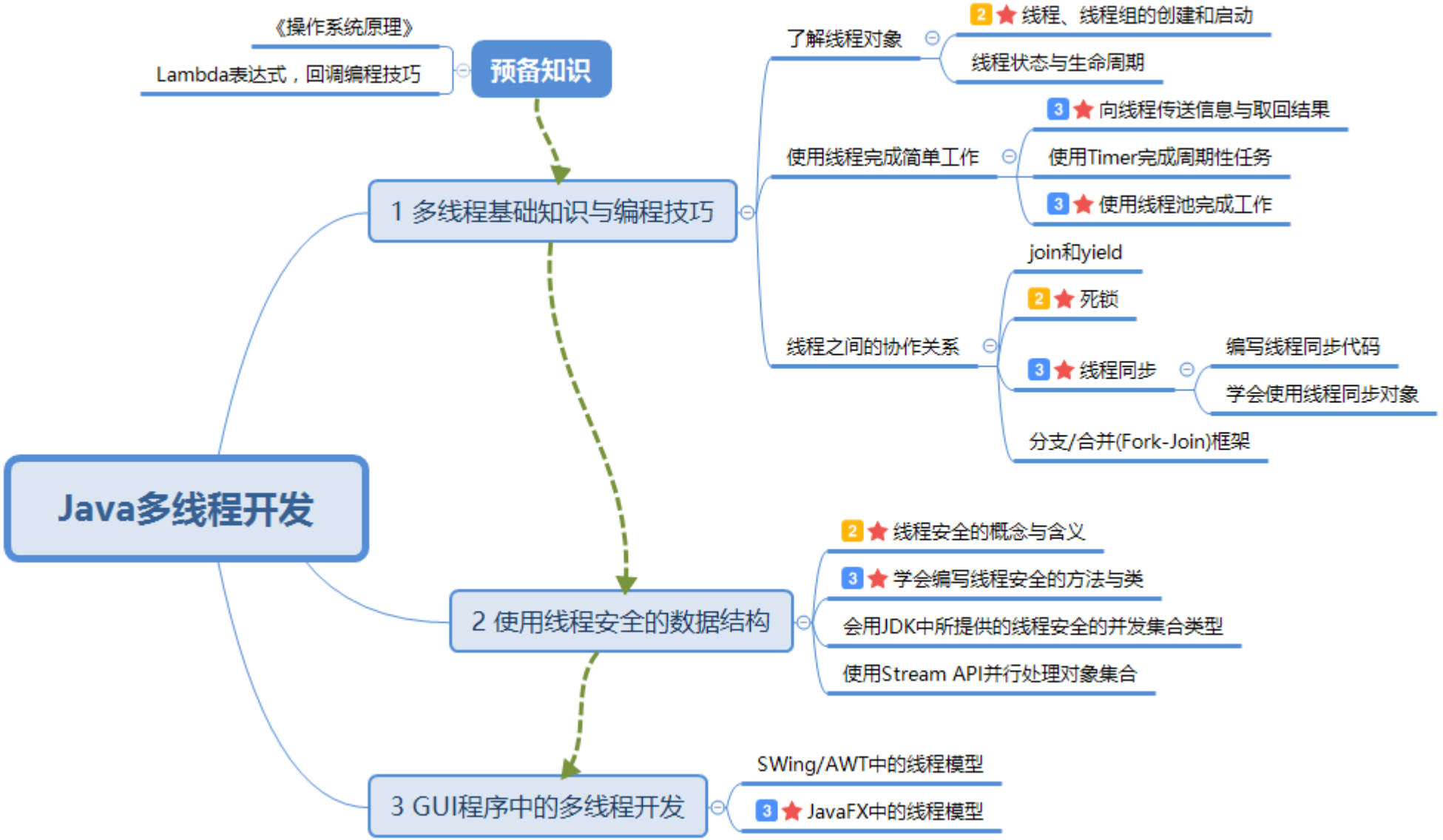
2.6 网络开发技术

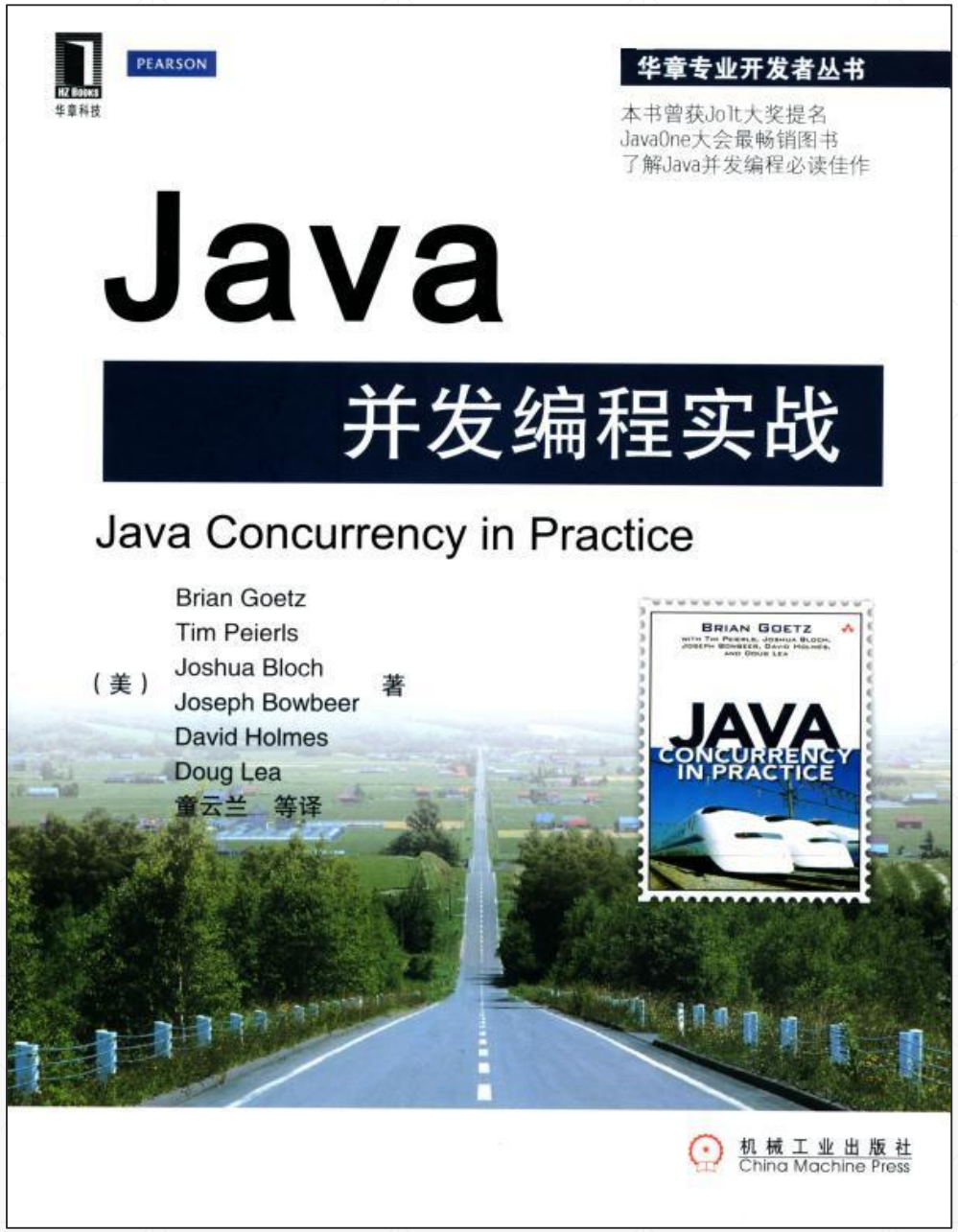

Java网络开发技术学习路线图
3 学习《计算机网络》大学教材
3 学习《操作系统》大学教材
打好理论基础
5 多线程基础
3 IO与流
4 Socket编程
5 NIO
JDK中与网络有关的核心技术
5 Netty
4 Vert.x
流行的网络开发框架
4 RxJava

Java网络开发技术关联图



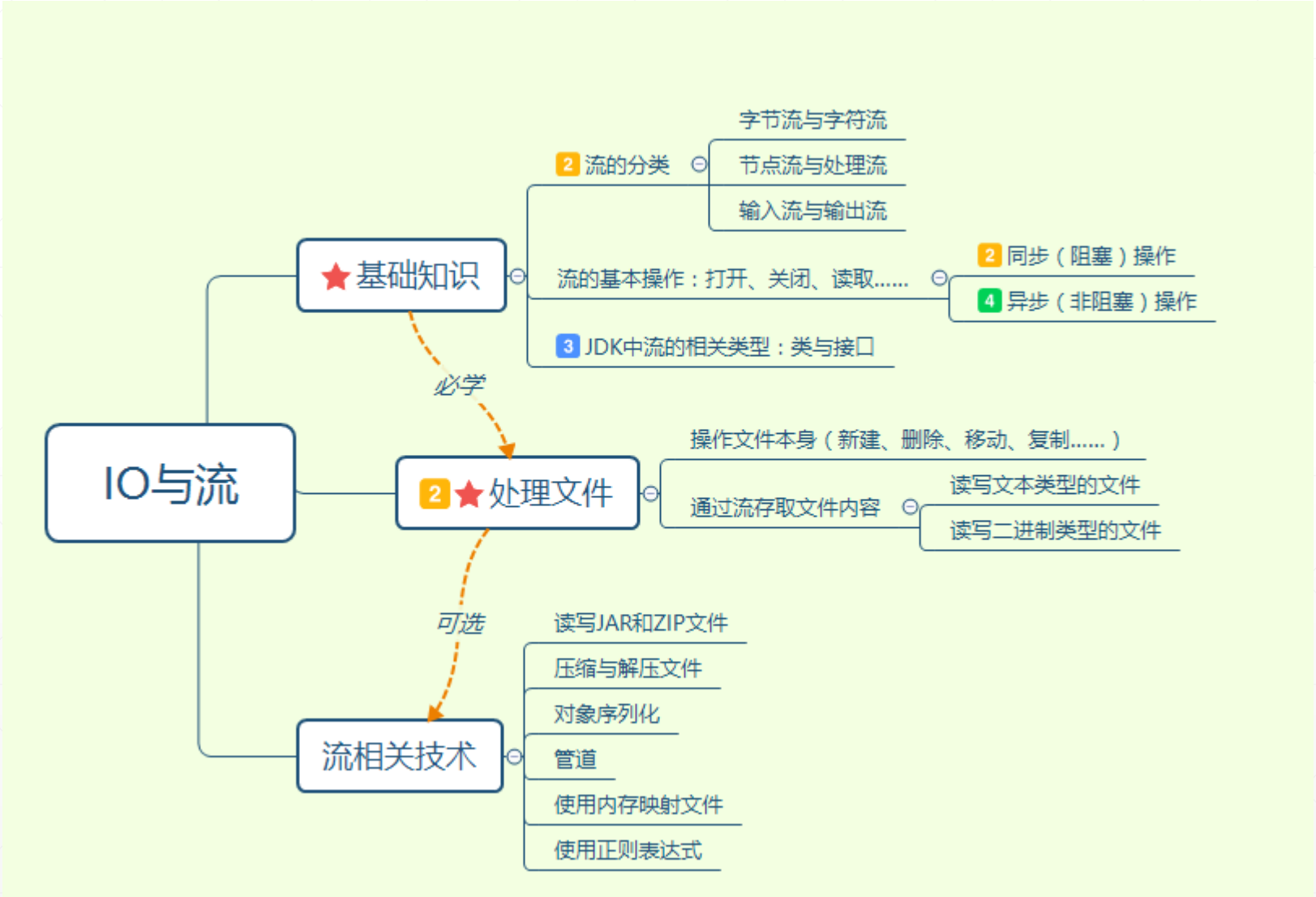
Java多线程开发知识关联图与学习路线





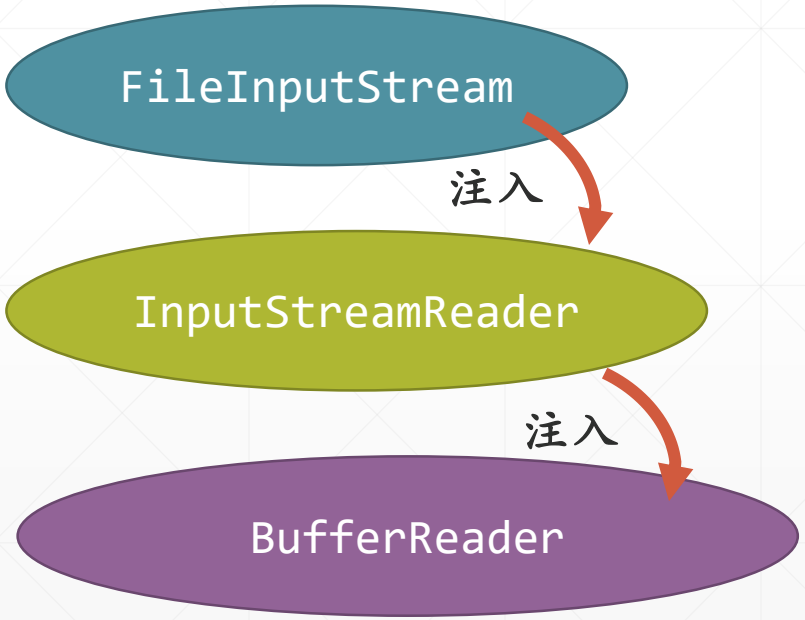
在多线程开发这块久负盛名的经典书籍，值得仔细研读。

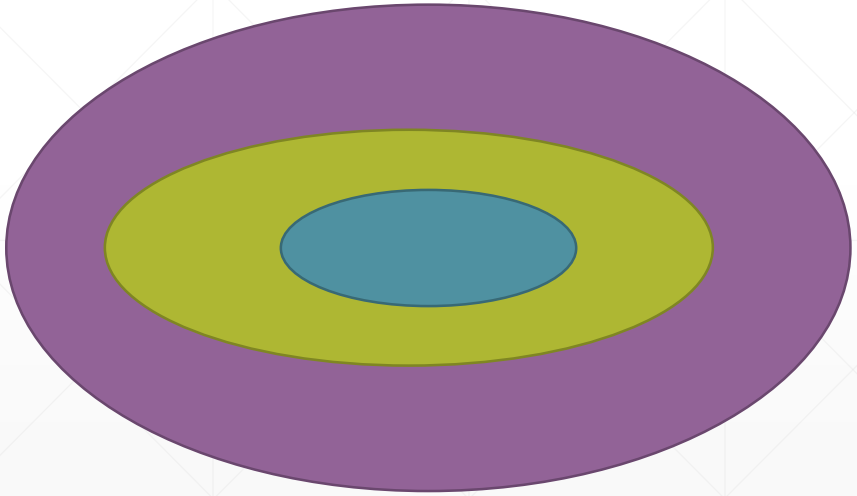
“IO与流”知识关联图与学习路线



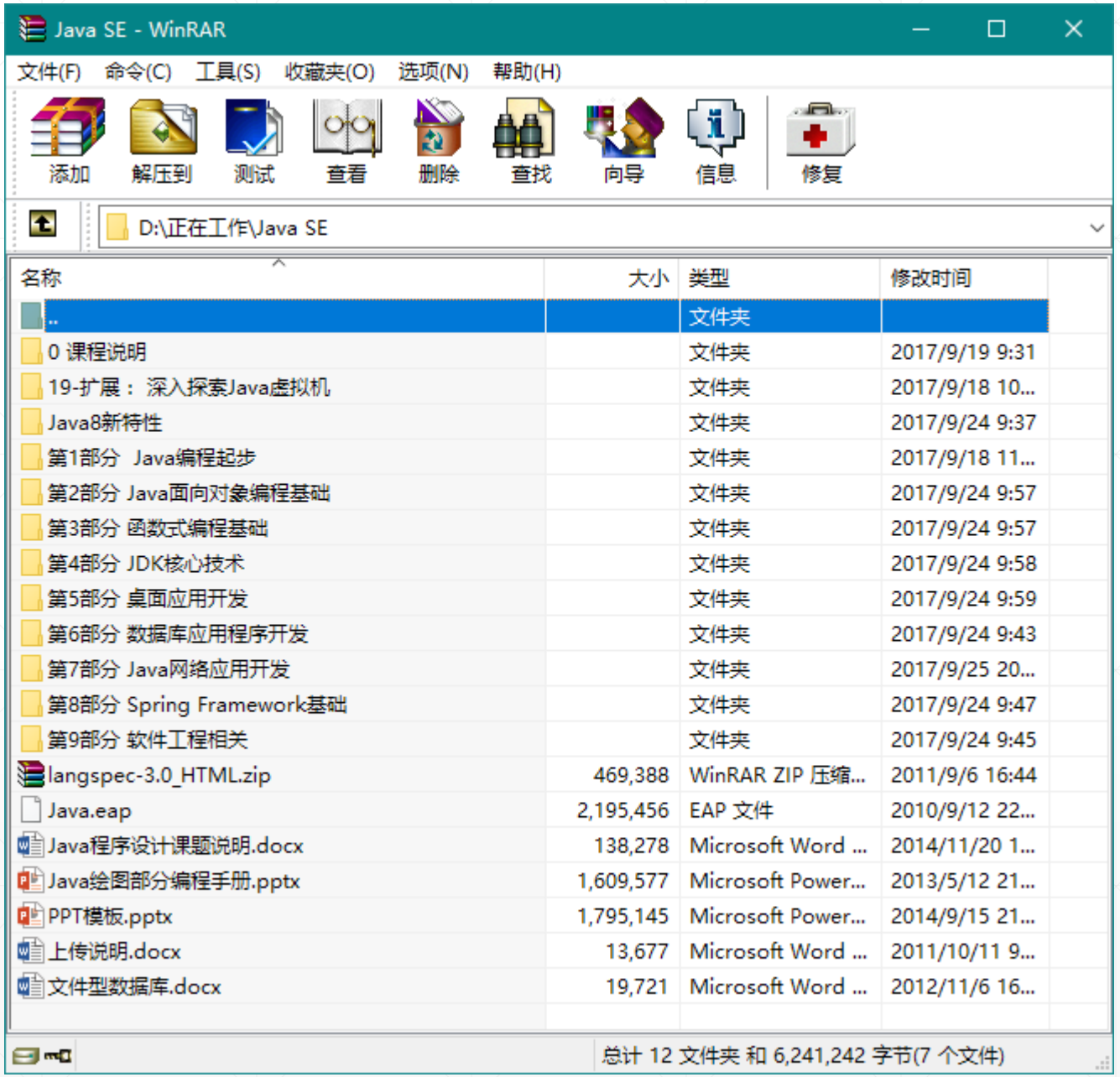
“节点流”与“处理流”

```
InputStream inputStream = new FileInputStream("input.txt");
Reader reader = new InputStreamReader(inputStream);
BufferedReader bufferedReader = new BufferedReader(reader);
String data = bufferedReader.readLine();
System.out.println(data);
```





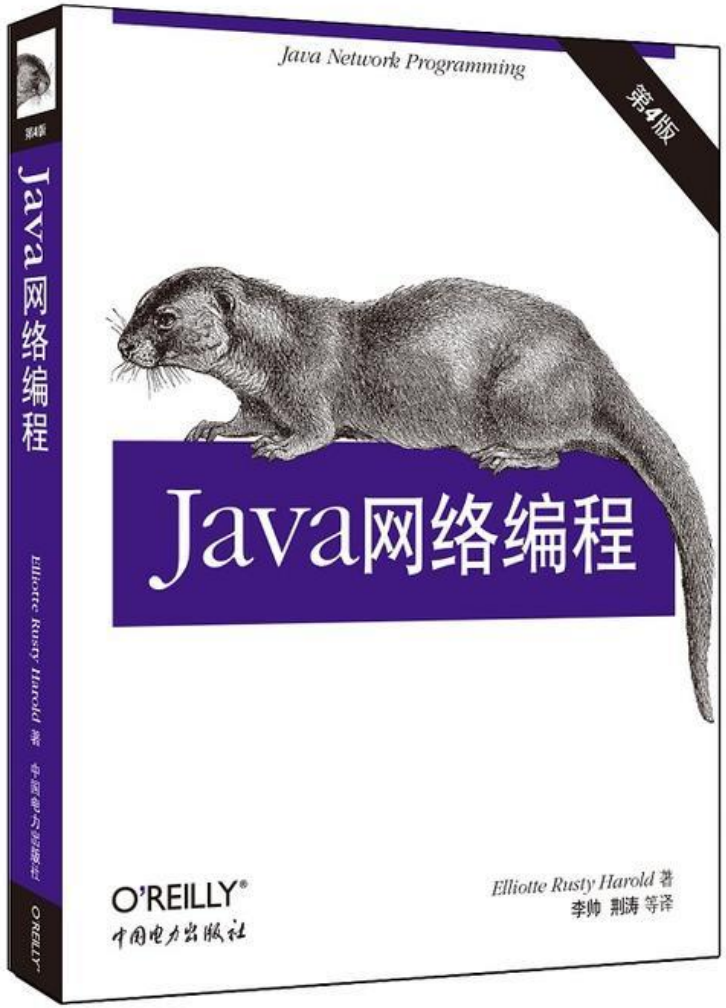
拥有“松花蛋”结构的流对象



IO流这块的刻意训练任务：

请克隆一个WinRAR，其特点：

- 压缩包中包容多个文件和文件，是一种递归的结构
- 包中的文件被压缩了
- 可以解压单个或多个文件
- 支持分卷压缩和解压
- 压缩和解压算法可以切换

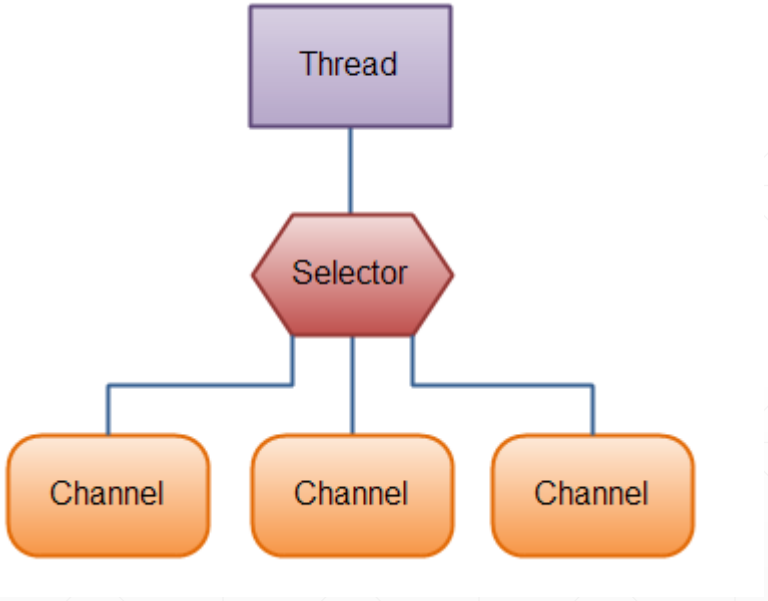


一本Java网络编程的入门级别书籍，讲得很细，但不够深入。适合于初学者。


Java经典网络开发技术
3 计算机网络原理
了解与掌握相关网络协议的特点与工作原理
1．IP、TCP和UDP
2．HTTP协议及上层协议
计算机如何互联成网？它们之间信息是如何传送的？
JDK基础技术
3 输入/输出流
4 多线程开发技术
学习Socket编程的前提
2 JDK中的辅助网络组件
InetAddress
URL与URI
NetworkInterface
URLConnection
.....
基于Socket的开发网络应用程序
IP地址与网络应用程序的端口
Socket的概念
5 Socket编程模型
创建与构建客户端与服务端Socket
连接的建立、关闭与管理
实现客户端与服务端的数据交换

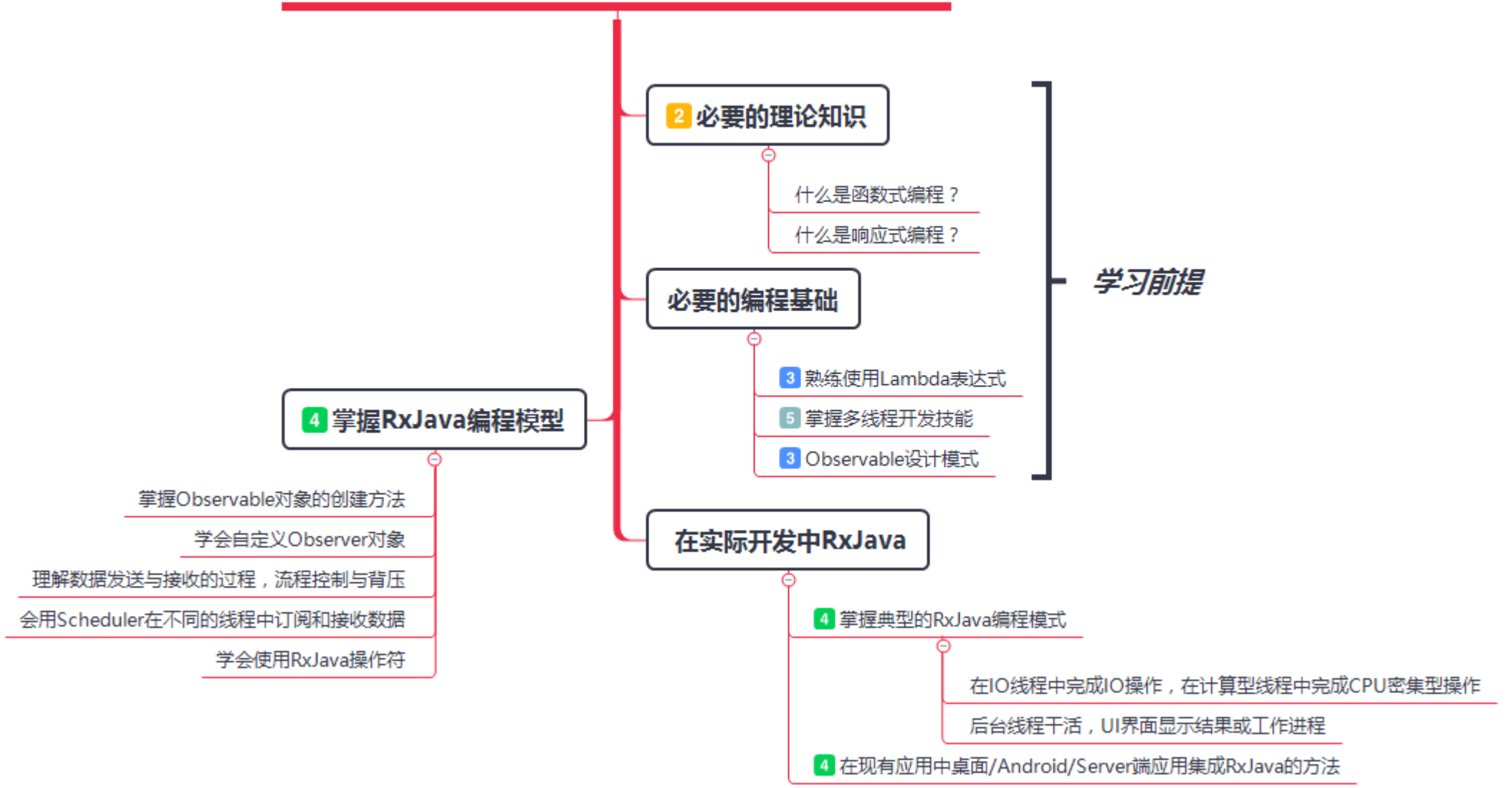
NIO三大组件之间的协作关系

Channel（通道）与Buffer（数据缓冲区）



Channel（通道）与
Selector（选择器）

★RxJava知识要点与学习路线



函数式编程（Functional Programming）核心概念

纯函数（Pure Function）

函数是一等公民（Function as First Class Citizen）

高阶函数（High Order Function ）

响应式编程（Reactive Programming）概念

事件驱动的（Event Driven）

可伸缩的（Scalable）

可恢复的（Resilient）

及时响应的（Responsive）

Netty是什么？

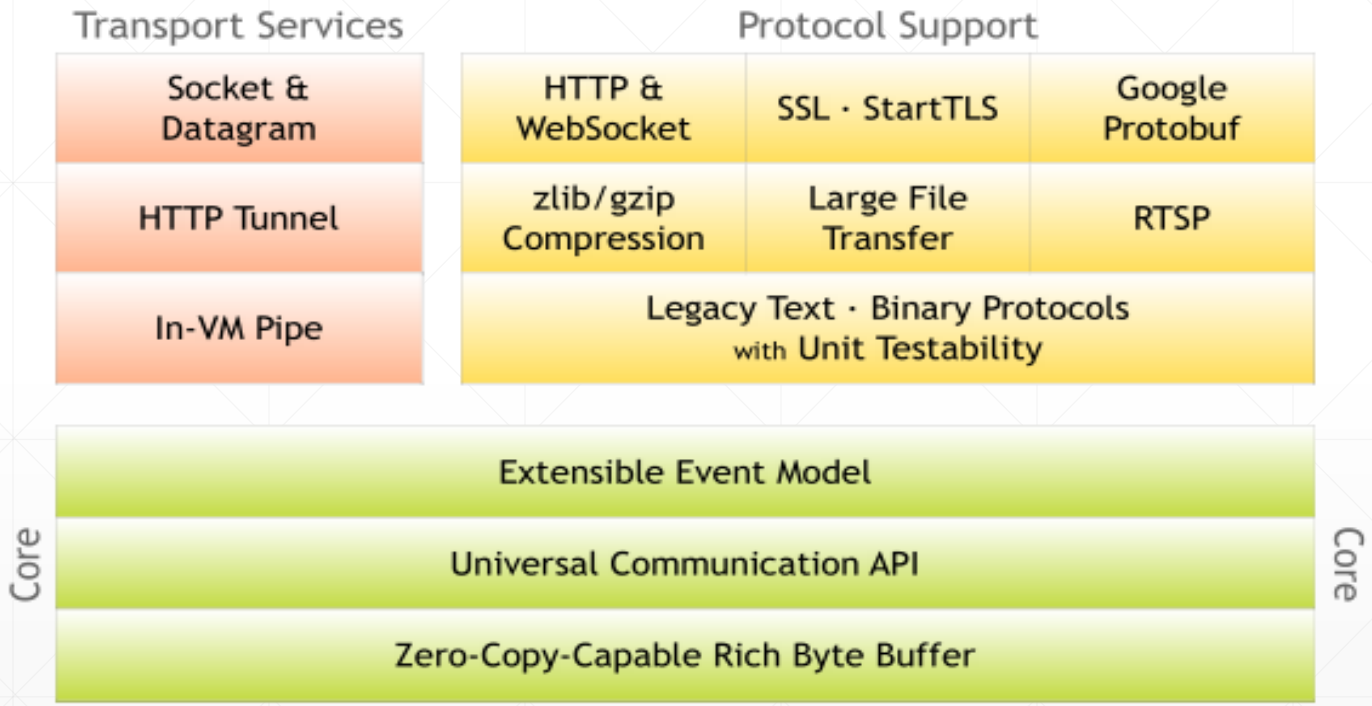
Netty is an asynchronous event-driven network application framework for rapid development of maintainable high performance protocol servers & clients.

——http://netty.io

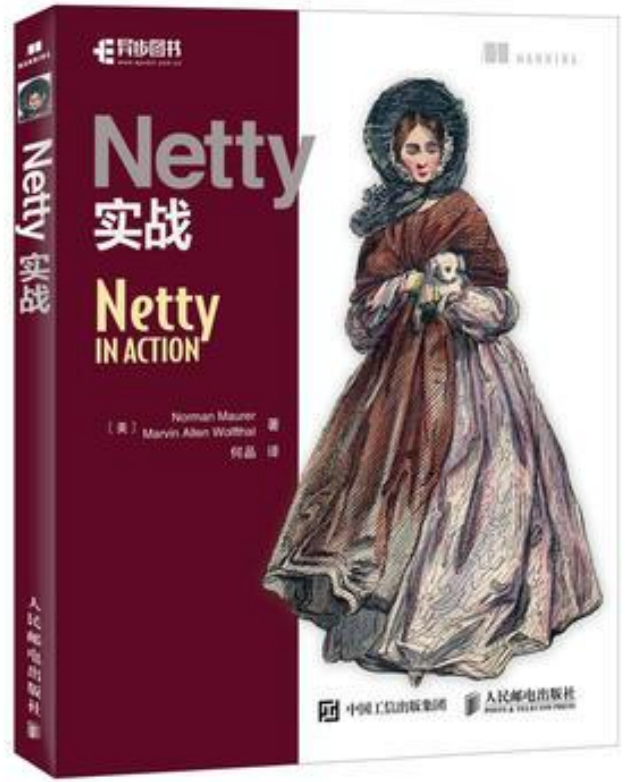
意译为中文

Netty是一个异步的、事件驱动的网络应用框架，可用于快速开发高性能的网络协议服务器和客户端。

Netty技术特性与相应技术书籍

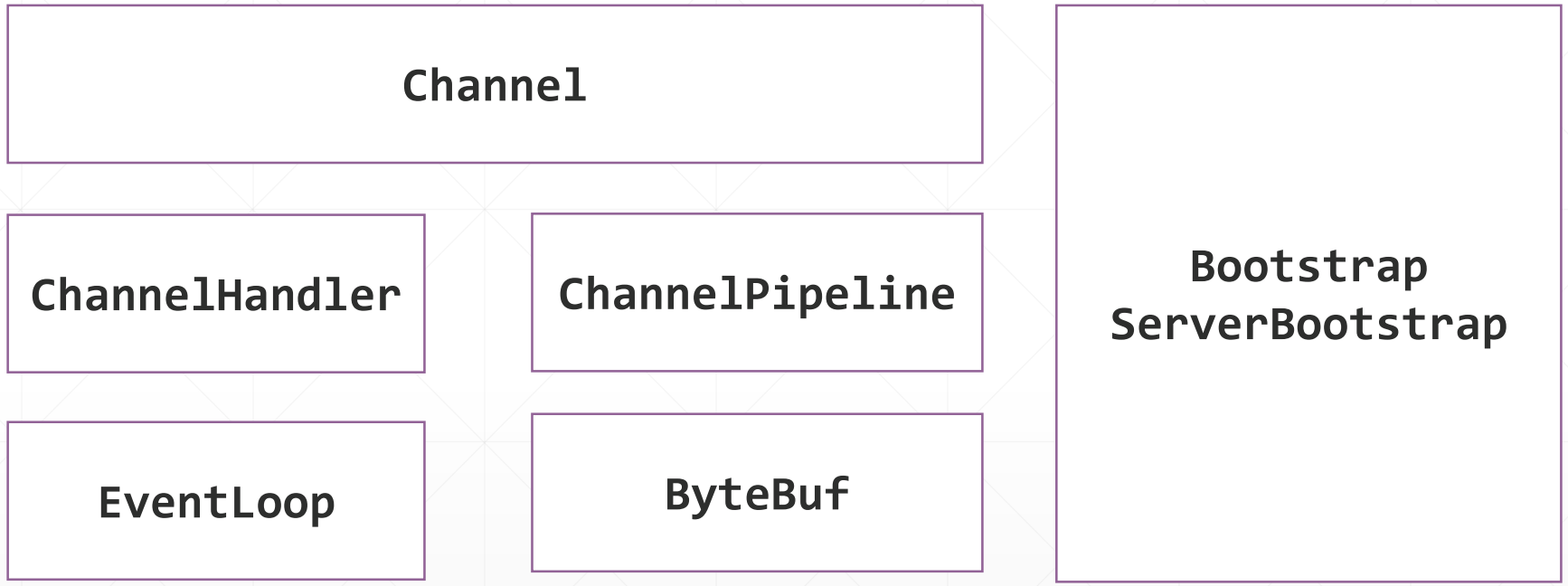


Netty官网总结的Netty技术特性



Netty专业书籍

Netty的核心组件

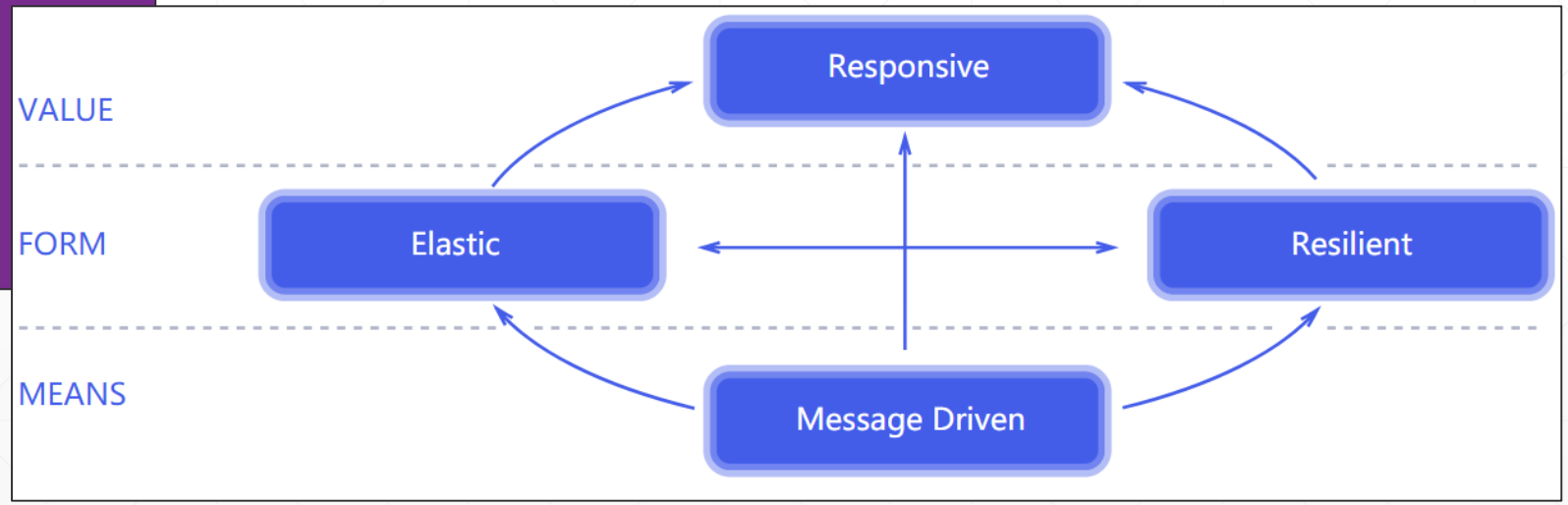


理解上述核心组件的职责与协作方式，是掌握Netty用法的关键。

Vert.x是什么？

http://vertx.io

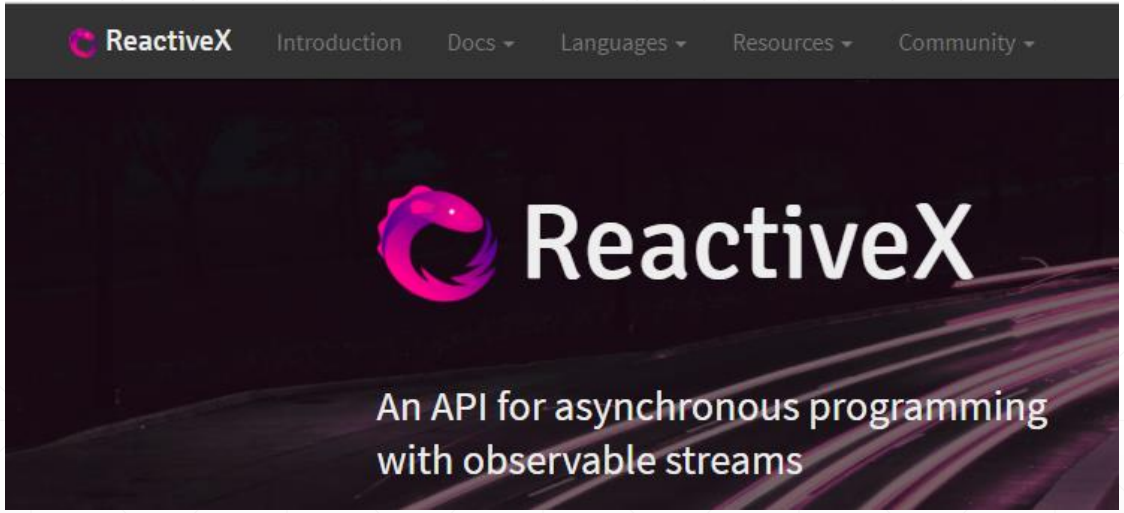
响应式系统宣言



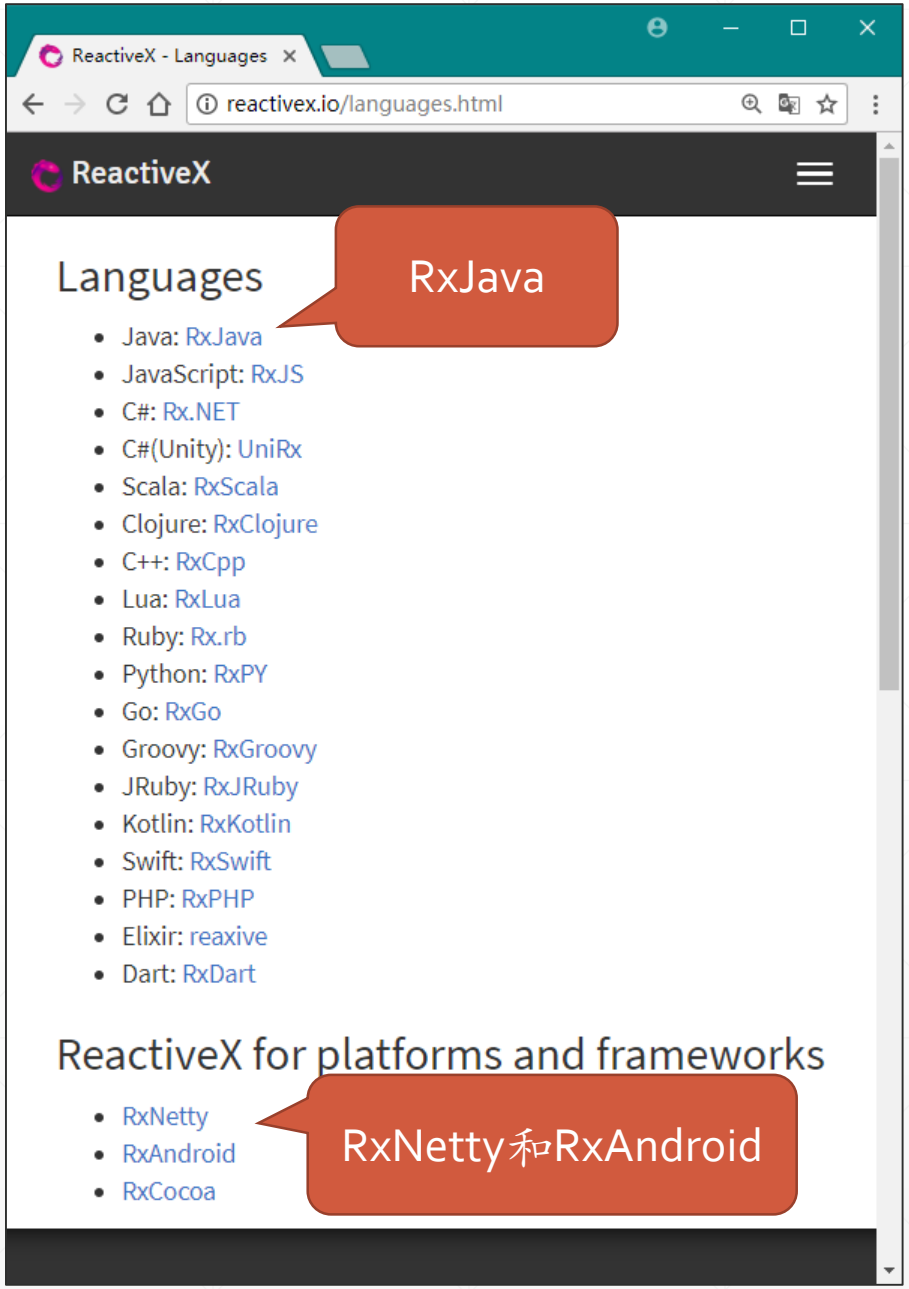
https://www.reactivemanifesto.org/

响应式编程开发框架

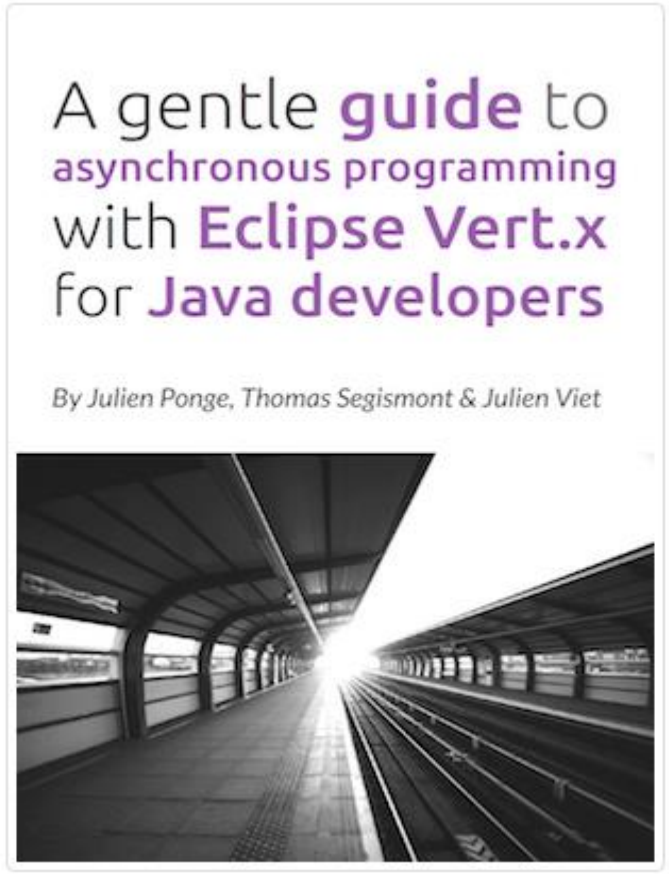
http://reactivex.io

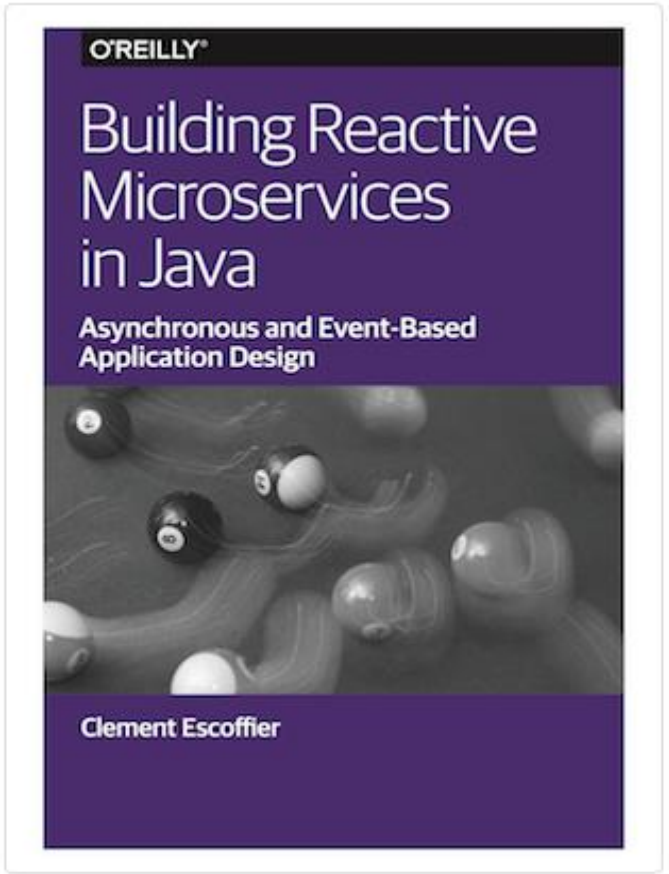


有多种编程语言构造了响应式编程框架

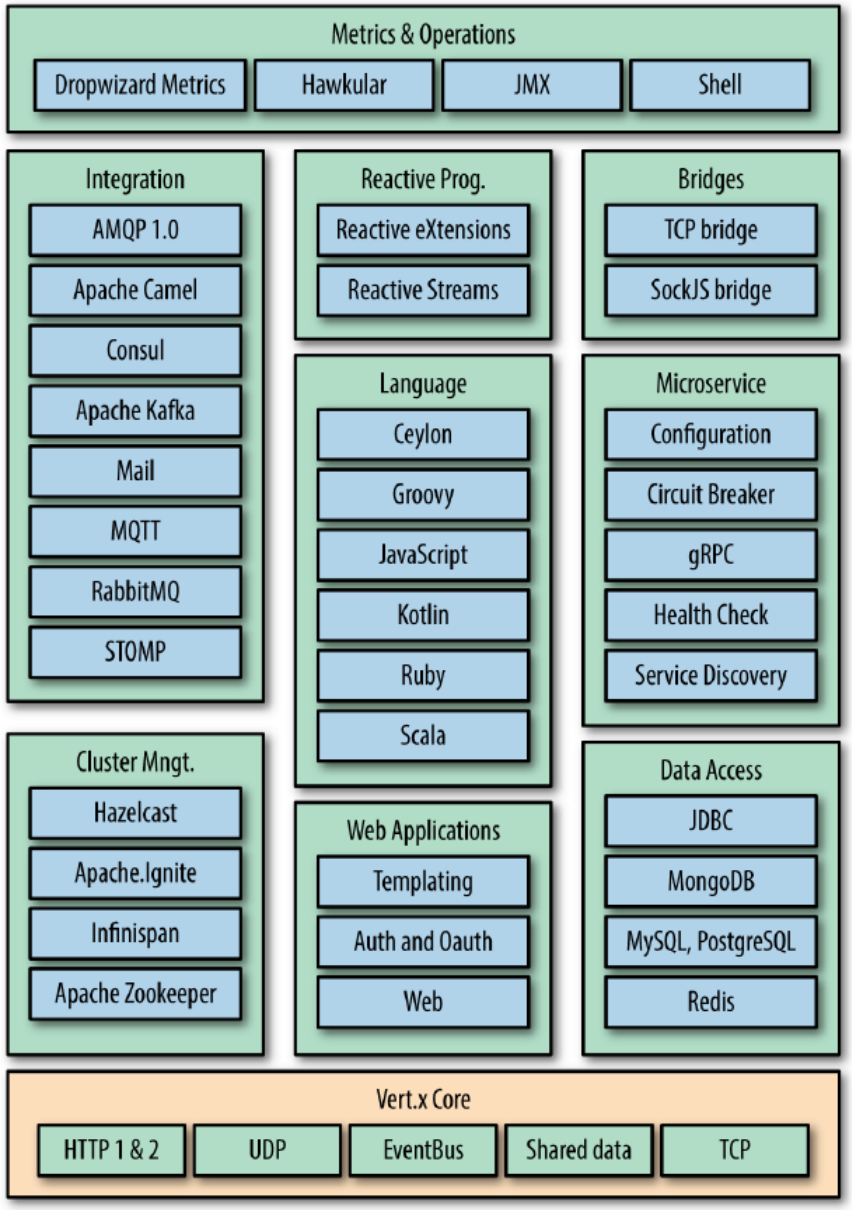


Vert.x官网上提供的两本电子书：

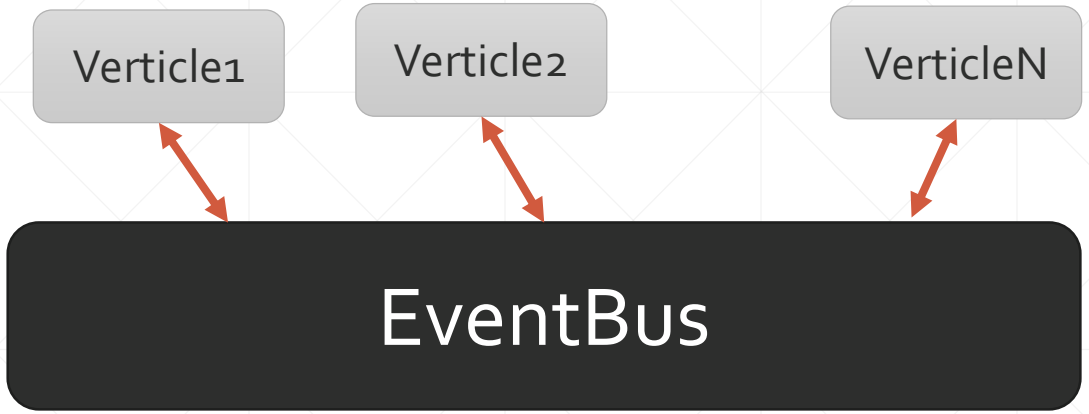




Vert.x生态系统



Vert.x编程模型



初学者只学核心组件就够了，其它的组件与模块依据工作需要再学。

Java网络开发技术“刻意练习”题

- 使用Socket编程实现两台计算机之间的一对一交谈。
- 实现一个支持网络对战的JavaFX网络小游戏，比如联网下五子棋。
- 编写一个文件共享服务器，所有用户都可以上传文件，也可以下载别人上传的文件，上传和下载支持断点续传。
- 编写一个基于TCP协议的网络聊天室，实现多人在线实时交流。
- 克隆QQ的部分功能，实现基于局域网的一对一聊天和群聊，包括客户端与服务器两块，采用Client/Server架构实现

本次Live到此结束

祝大家学习进步！