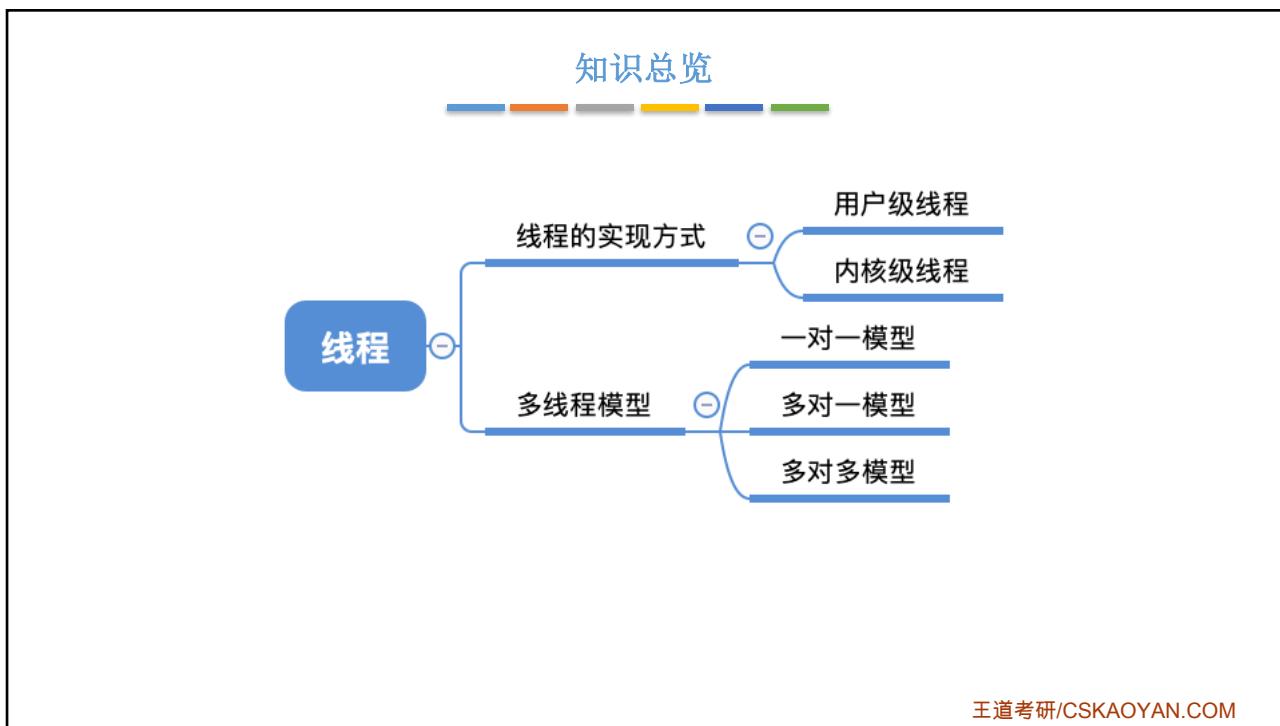


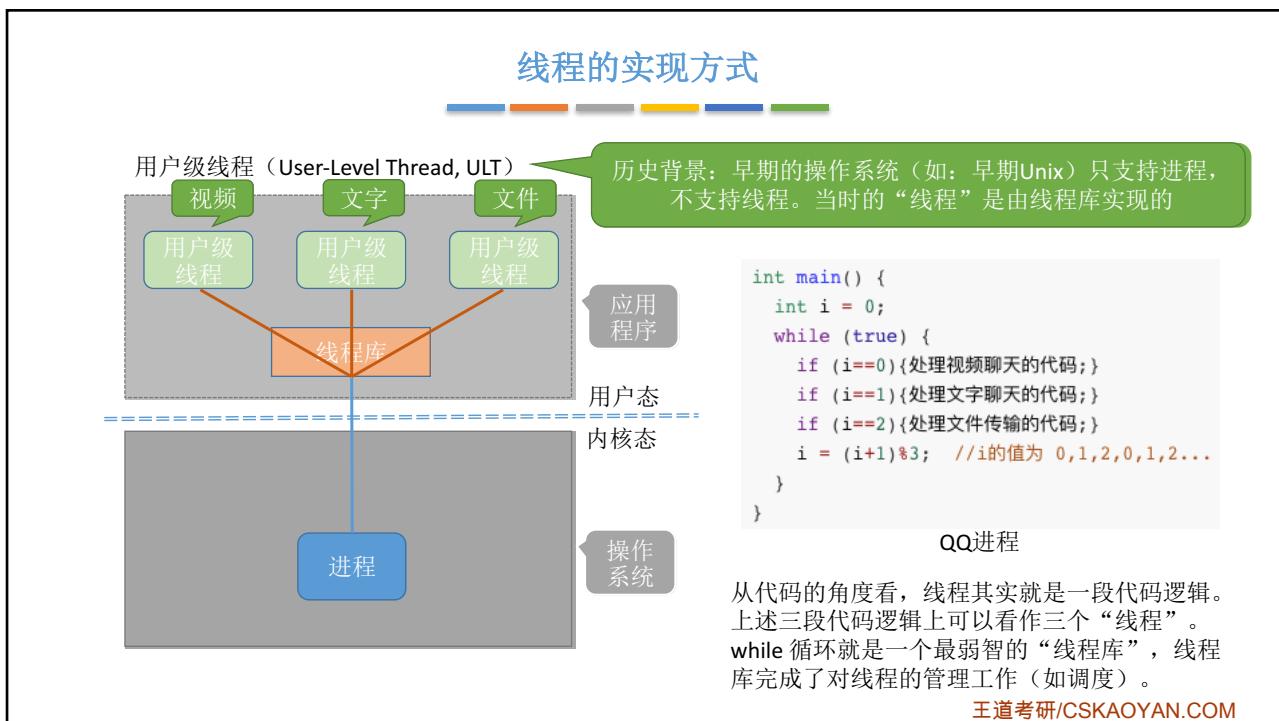
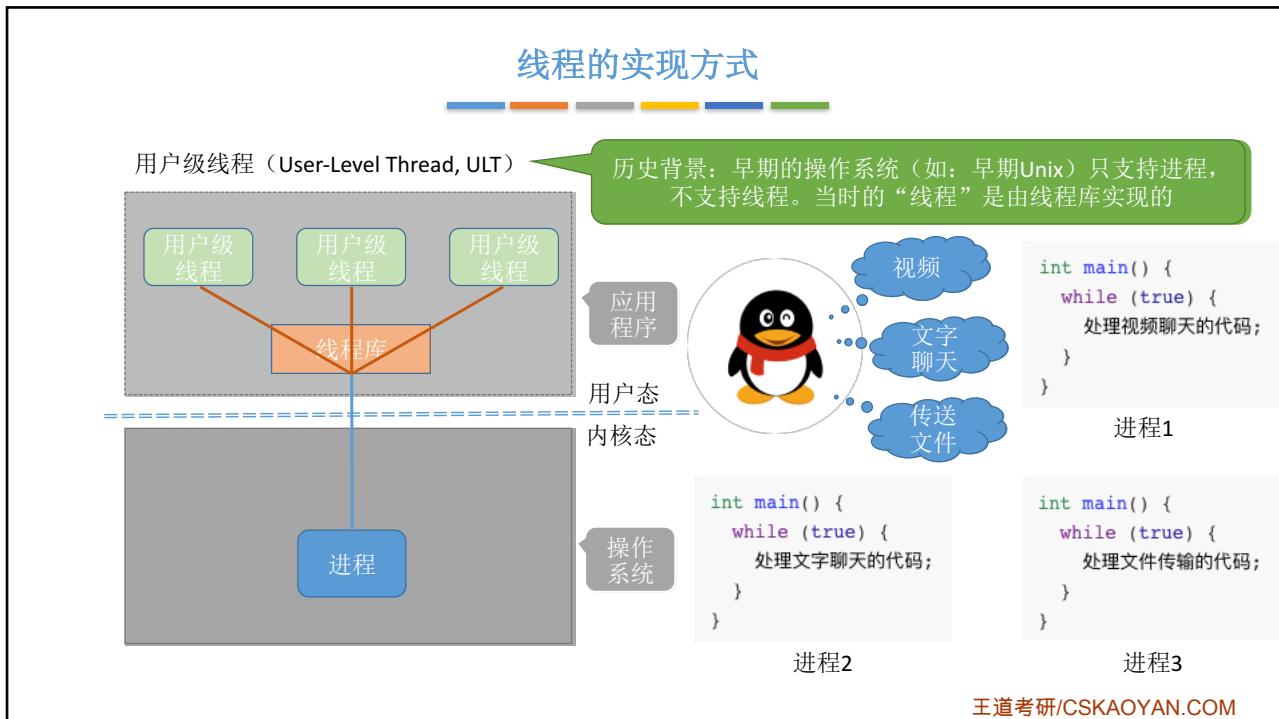
本节内容

线程的实现方式

多线程模型

王道考研/CSKAOYAN.COM





线程的实现方式

用户级线程 (User-Level Thread, ULT)

历史背景：早期的操作系统（如：早期Unix）只支持进程，不支持线程。当时的“线程”是由线程库实现的。

很多编程语言提供了强大的线程库，可以实现线程的创建、销毁、调度等功能。

应用程序

用户态
内核态

操作
系统

进程

线程库

1. 线程的管理工作由谁来完成？
 2. 线程切换是否需要CPU变态？
 3. 操作系统是否能意识到用户级线程的存在？
 4. 这种线程的实现方式有什么优点和缺点？

王道考研/CSKAOYAN.COM

线程的实现方式

用户级线程 (User-Level Thread, ULT)

1. 用户级线程由应用程序通过线程库实现，所有的线程管理工作都由应用程序负责（包括线程切换）
 2. 用户级线程中，线程切换可以在用户态下即可完成，无需操作系统干预。
 3. 在用户看来，是有多个线程。但是在操作系统内核看来，并意识不到线程的存在。
 “用户级线程”就是“从用户视角看能看到的线程”

4. 优缺点
 优点：用户级线程的切换在用户空间即可完成，不需要切换到核心态，线程管理的系统开销小，效率高
 缺点：当一个用户级线程被阻塞后，整个进程都会被阻塞，并发度不高。多个线程不可在多核处理器上并行运行。

应用程序

用户态
内核态

操作
系统

进程

线程库

用户级线程

用户级线程

用户级线程

王道考研/CSKAOYAN.COM

