# Threading Programming Guide

[원문 링크]

## 문서 읽기 순서

- 빨간색
  - 초보에게 꼭 필요한 개념입니다. 이해하기 쉬운 편입니다.
- 녹색
  - 초보에게 꼭 필요한 개념입니다. 이해하기 어려운 편입니다.
- 주황색
  - 특정 상황에만 필요하거나 초보에게 당장은 필요하지는 않은 개념입니다. 이해하기 쉬운 편입니다.
- 파란색
  - 특정 상황에만 필요하거나 초보에게 당장은 필요하지는 않은 개념입니다. 이해하기 어려운 편입니다.
- 갈색
  - 지금 당장 볼 필요는 없고, 필요한 경우에 찾아올 수 있도록 이런 내용이 있다는 것만 알아두면 됩니다.

---

## Introduction

> 이 문서는 스레드의 개념과 애플리케이션 설계에서의 역할을 설명하고, 스레딩 기술과 사용방법 그리고 보조 스레드에서 이벤트 처리 루프를 관리하는 방법과 동기화, macOS 및 iOS의 스레드 안정성에 관해 설명합니다. 부록으로 스레드에 관련된 용어들을 설명합니다.

## About Threaded Programming

> 스레드에 관해 설명하고 스레드 용어 및 단일 스레드를 효율적으로 사용할 수 있는 방법과 스레딩 기술 및 스레드 설계 팁에 관해 설명합니다.

- What Are Threads?
  - 스레드의 개념과 응용프로그램 내에서 스레드 활용 방법에 관해 설명합니다.

- Threading Terminology
  - `스레드`, `프로세스`, `태스크(작업)` 용어를 정의합니다.
- Alternatives to Threads
  - 스레드를 조금더 효율적으로 사용할 수 있는 대체 기술에 관해 설명 합니다.
- Threading Support
  - 스레딩 기술, 스레드 실행 루프, 스레드 동기화, 스레드 간 통신에 관해 설명 합니다.
- Design Tips
  - 스레드 생성, 유휴 스레드 관리, 스레드 공유 데이터, 스레드와 사용자 인터페이스, 응용프로그램 종료 시점의 스레드 관리, 스레드 예외처리, 스레드의 완전한 종료, 라이브러리 내의 스레드 사용방법 등의 팁을 제공합니다.

# Thread Management

> 스레드를 사용 시 메모리상의 사용 및 성능비용에 관해 설명하고 스레드를 생성하는 방법과 스레스의 속성에 관해 설명합니다. 스레드의 진입점과 종료 시점에 관해서도 알아봅니다.

- Thread Costs
  - `커널 데이터 구조`, `스택 공간`, `생성 시간`에서 스레드 생성 비용(메모리, 시간)에 관해 설명합니다.
- Creating a Thread
  - `NSThread`, `POSIX`, `NSObject`를 이용한 스레드 생성 방법에 관해 설명합니다.
- Configuring Thread Attributes
  - 스레드를 생성한 후 변경할 수 있는 속성들을 설명합니다.
- Writing Your Thread Entry Routine
  - 스레드 진입점을 정의하고 `오토릴리즈 풀`, `예외처리`, `실행 루프`를 설정하는 방법에 관해 설명합니다.
- Terminating a Thread
  - 스레드를 안전하게 종료하는 방법과 강제로 종료하는 방법에 관해 설명합니다.

# Run Loops

> 실행 루프 객체를 이용하여 스레드의 실행 루프를 구성하고 관리하는 방법에 관해 설명합니다.

- Anatomy of a Run Loop

    - 루프 모드를 실행 할 때 특정 모드를 지정하여 스레드의 이벤트들을 어떻게 전달하는지, 이벤트가 발생할 때마다 옵저버(관찰자)를 통해 처리하는 방법에 관해 설명합니다.

- When Would You Use a Run Loop?
    - 언제 실행 루프가 필요한지에 관해 설명합니다.
- Using Run Loop Objects
    - 실행 루프의 객체를 참조하여 소스를 구성하고 시작, 종료에 관해 설명합니다.
- Configuring Run Loop Sources
    - 다양한 유형의 입력 소스를 설정하는 방법에 관해 설명합니다. ( `커스텀` , `타이머` , `포트 기반` )

# Synchronization

> 응용프로그램 내에 여러 스레드가 있으면 여러 스레드가 실행되면서 공유된 리소스를 액세스하는 것에 문제가 발생할 수 있습니다. 이 문제를 스레드 간의 동기화 도구 등을 이용해 프로그램 리소스에 안전하게 액세스하는 방법을 설명합니다.

- Synchronization Tools
    - 다른 스레드가 예기치 못한 상태에서 데이터를 변경하지 못하도록, 동기화의 문제가 없도록 응용프로그램 설계 및 동기화 도구를 통해 설명합니다.
- Synchronization Costs and Performance
    - 스레드 동기화 설계 시 성능과 비용에 관해 장단점을 설명합니다.
- Thread Safety and Signals
    - 응용프로그램의 내의 스레드의 안정성 및 신호처리에 관해 설명합니다.
- Tips for Thread-Safe Designs
    - 스레드의 안전성과 성능 사이의 균형을 찾는 방법에 관해 설명합니다.
- Using Atomic Operations
    - 원자 연산을 이용해 두 스레드 간의 동기화를 효과적으로 수행할 방법에 관해 설명합니다.
- Using Locks
    - 스레드 프로그래밍을 위한 기본적인 동기화 도구인 락(잠금)에 관해 설명합니다. ( `POSIX Mutex` , `NSLock` , `@synchronized` , `NSConditionLock` , `NSDistributedLock` )

- Using Conditions
    - 스레드의 또 다른 동기화 작업인 조건 작업에서 뮤텍스와의 차이점과 사용방법에 관해 설명합니다. ( `NSCondition` , `POSIX` )

# Thread Safety Summary

> 부록에선 macOS 및 iOS의 프레임워크에서 고급 스레드 안정성에 관해 설명합니다.

- Cocoa
    - 코코아에서 스레드를 사용하기 위한 지침과 안정성에 관해 설명합니다.
- Core Foundation
    - 코어 파운데이션에서 스레드를 사용할 시 주의할 점에 관해 설명합니다.

# Glossary

> 스레드에 관련된 용어설명이 나열되어 있습니다.