**Flags и QoS**

**Флаги (Flags)**

Флаги представляют собой дополнительные параметры, которые можно передать при создании объекта DispatchWorkItem.

Они контролируют различные аспекты выполнения блока, такие как использование барьеров, наследование контекста выполнения, использование уровней QoS и т. д.

Флаги позволяют точно настраивать поведение выполнения задачи, но часто используются для специфических сценариев или требований.

**Уровни качества обслуживания (QoS)**

Уровни качества обслуживания (QoS) определяют приоритет выполнения задачи в очереди диспетчеризации.

Они представлены классом DispatchQoS, который определяет четыре уровня приоритета: User-Interactive, User-Initiated, Utility и Background.

Уровни QoS помогают GCD принимать решения о том, как эффективно распределить ресурсы процессора и обеспечить отзывчивость приложения.

**Разница**

Флаги управляют конкретными аспектами выполнения задачи, такими как использование барьеров, наследование контекста выполнения и т. д.

Уровни качества обслуживания (QoS) определяют приоритет выполнения задачи в контексте всей системы, позволяя GCD эффективно управлять ресурсами процессора и обеспечить отзывчивость приложения.

Вместе флаги и уровни QoS предоставляют различные инструменты для управления выполнением задач в GCD, обеспечивая гибкость и эффективность в многопоточной среде.

**Параметры flags:**

*.barrier*: Этот флаг указывает, что блок является барьерным. Барьерные блоки гарантируют, что все предыдущие блоки в данной очереди будут завершены до выполнения барьерного блока, а все последующие блоки будут ожидать завершения барьерного блока перед выполнением. Это часто используется для обеспечения синхронизации при доступе к общим ресурсам из нескольких потоков.

*.detached*: Этот флаг указывает, что блок является независимым от родительской задачи. Он может быть полезен, если вы хотите, чтобы выполнение блока не блокировало родительскую задачу.

*.assignCurrentContext*: Этот флаг указывает, что блок должен использовать текущий контекст выполнения. Это означает, что блок будет выполняться в том же контексте, что и место, где он был создан.

*.inheritQoS*: Этот флаг указывает, что блок должен наследовать уровень качества обслуживания (QoS) от родительской задачи.

*.noQoS:* Этот флаг указывает, что блок не должен использовать уровень качества обслуживания (QoS) при выполнении.

*.enforceQoS*: Этот флаг указывает, что блок должен применять уровень качества обслуживания (QoS) к очереди диспетчеризации.

*.assignCurrentContext*: Этот флаг указывает, что блок должен использовать текущий контекст выполнения.

*.inheritQoS*: Этот флаг указывает, что блок должен наследовать уровень качества обслуживания (QoS) от родительской задачи.

*.unspecified*: Этот флаг указывает, что должен использоваться неопределенный уровень качества обслуживания (QoS). Это значение обычно используется по умолчанию.

**Классы QoS:**

*User-Interactive (userInitiated) Пользовательский интерфейс*: Высший уровень приоритета, предназначен для задач, которые требуют немедленного отклика пользователя, таких как обновление пользовательского интерфейса или анимации.

*User-Initiated (userInitiated*) *Пользовательская инициатива*: Средний уровень приоритета, подходит для задач, запущенных пользователем и требующих интерактивного отклика, но не так срочно, как User-Interactive. Это может включать загрузку данных из сети или обработку пользовательского ввода.

*Utility утилита:* Этот класс используется для задач, которые могут быть отложены без ущерба для пользовательского опыта. Задачи этого класса выполняются с низким приоритетом. Примеры задач: сетевые запросы в фоновом режиме, обработка данных в фоне.

*Background (background) фоновый*: Низкий уровень приоритета, используемый для задач, которые могут быть выполнены в фоновом режиме без влияния на пользовательский опыт, например, синхронизация данных или резервное копирование.

*Unspecified (Неопределенный)*: класс используется, когда QoS не определен явно. Обычно это случай, когда QoS не указан в явном виде или информация о нем утрачена. Он имеет наименьший приоритет среди всех классов QoS. Примеры: задачи, для которых не указан QoS, или задачи, у которых QoS был утрачен.

*Default (По умолчанию)*: Этот класс используется, когда явно не указан другой класс QoS. Он используется для обычных задач, которые не имеют специальных требований к приоритету. Обычно операционная система ставит задачи этого класса в очередь после задач класса User-initiated и до задач класса Utility. Примеры: стандартные вычисления, обработка данных.