**Барьеры отправки**

**Определение**

Механизм в GCD, который обеспечивает защиту доступа к общим данным при использовании параллельных очередей. Они позволяют задавать определенные точки в выполнении кода, где выполнение задач становится последовательным, чтобы предотвратить одновременный доступ к общим данным из нескольких потоков.

**Механизм работы**

Барьеры отправки используются с параллельными очередями, чтобы разделить код на две части: код до барьера и код после барьера.

При достижении барьера отправки все задачи, добавленные в очередь до барьера, выполняются последовательно, прежде чем начнется выполнение задач, добавленных после барьера.

Это гарантирует, что только одна задача выполняется в очереди после барьера в любое время, что обеспечивает безопасность доступа к общим данным.

**Создание**

Для создания барьера отправки используется метод async(flags:execute:) объекта DispatchQueue, где параметр flags указывает, что задача является барьером отправки.

В Swift барьер отправки создается с помощью атрибута .barrier.

1. concurrentQueue.async(flags: .barrier) {
2. // Код после барьера отправки
3. }

**Использование барьеров отправки**

Барьеры отправки часто используются для обновления или модификации общих данных, таких как массивы или словари, в многопоточной среде, где безопасность доступа к данным очень важна.

Примеры использования барьеров отправки включают в себя добавление, удаление или изменение элементов в общей коллекции данных.

Барьеры отправки могут привести к замедлению выполнения вашего кода, поскольку они приводят к последовательному выполнению задач в параллельной очереди.

**Пример использования барьеров отправки**

1. let concurrentQueue = DispatchQueue(label: "com.example.concurrentQueue", attributes: .concurrent)
2. var sharedArray = [Int]()
3. // Добавление элементов в общий массив с использованием барьера отправки
4. func addElement(element: Int) {
5. concurrentQueue.async(flags: .barrier) {
6. sharedArray.append(element)
7. }
8. }
9. // Удаление элементов из общего массива с использованием барьера отправки
10. func removeElement(at index: Int) {
11. concurrentQueue.async(flags: .barrier) {
12. sharedArray.remove(at: index)
13. }
14. }