<https://developer.apple.com/documentation/uikit/app_and_environment/managing_your_app_s_life_cycle>

<https://developer.apple.com/documentation/uikit>

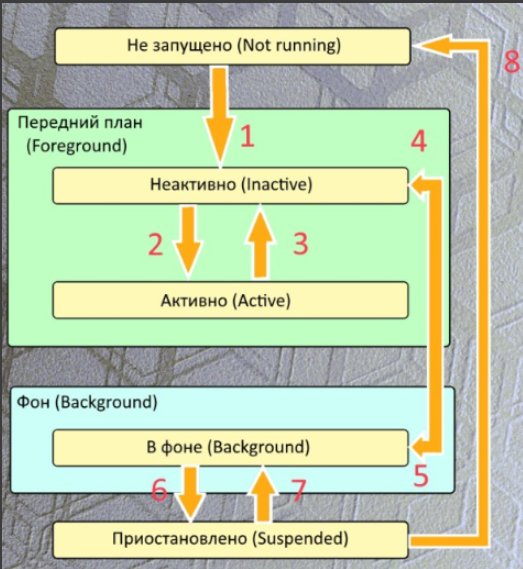
Понимание того, как устроено и работает приложение очень важно для более осмысленной работы с ним.

Когда пользователь только что включил свой девайс, приложения не работают, кроме тех, которые принадлежат операционной системе. После того, как пользователь коснется значка приложения - приложение запускается. Приложение и общие библиотеки, которые оно должно выполнить, будут загружаются в память. В итоге приложение начинает выполнение.

**Жизненный цикл приложения** - это совокупность определенных стадий/состояний и переходов между ними.

**Состояния приложения**

В любой момент времени приложение находится в каком-либо из перечисленных ниже состояний:



Система меняет состояния приложения в ответ на происходящие события. Например, когда пользователь нажимает кнопку Home, или поступает входящий вызов, или что либо еще — приложения в ответ на все это меняют свое состояние.

* **Not Running** - Приложение не запущено, либо запущенно но прервано системой.
* **Inactive** - Приложение работает, но в настоящий момент ничего не делает (на этом этапе выполняется код программы, но не обрабатываются события интерфейса пользователя (интерфейс не отображается, касания не обрабатываются и.т.п.)). Приложение, как правило, остается в этом состоянии очень мало времени и переходит в другое состояние
* **Active** - Нормальный режим работы приложения на переднем плане - выполняется код и обрабатываются все события UI.
  + - Если пользователь запустит другое приложение, то текущее приложение перейдет в состояние **Inactive** и затем в состояние **Background**.
* **Background** - Приложение находится в фоне, но работает. Большинство приложений входят в это состояние на короткое время и позже приостанавливаются. Но если необходимо дополнительное время для работы в бэкграунде, приложение может оставаться в этом состоянии. В этом состоянии коды выполняется ограниченное время, и не обрабатываются события UI. Нужно понимать, что именно в этом состоянии можно заставить приложения получить, допустим, из интернета самые свежие обновления и дать их приложению, чтобы когда пользователь вернул приложение в Foreground он мог увидеть эту информацию.
* **Suspended** - Приложение работает в фоне, но не выполняет никакой код. Система перемещает приложение в это состояние автоматически и не предупреждает об этом. При условии малого количества памяти, система может не предупреждая закрыть приложения в этом состоянии для освобождения памяти.

Текущее состояние приложения определяет, какие системные ресурсы ему доступны. Поскольку активные приложения отображаются на экране и должны реагировать на пользовательские взаимодействия, они имеют приоритет при использовании системных ресурсов. Фоновые приложения не отображаются на экране и поэтому имеют более ограниченный доступ к системным ресурсам и получают ограниченное время выполнения.

Приложение должно быть готово к завершению в любое время. Завершение — это нормальная часть жизненного цикла. Система обычно выключает приложения, для очищения памяти и подготовки к запуску других приложений, которые запущены пользователем, но система так же может выключить приложения , которые некорректно или не отвечающим на события своевременно.

Suspended приложения не получают уведомления о завершении. Система убивает процесс и восстанавливает соотвествующую память.

Также пользователь может завершить приложение явно с помощью многозадачного пользовательского интерфейса. Инициированное пользователем прекращение имеет тот же эффект, что и прекращение приостановленного приложения. Процесс приложения убит, и уведомление не отправляется в приложение.

### Методы перехода между состояниями

Большинство переходов между состояниями обеспечивается соответствующими методами, которые вызываются или будут реагировать на некоторые из вышеуказанных состояний:

Запуск:

* **application:willFinishLaunchingWithOptions**-  метод вызывается для инициализации структур данных приложения и выполнения одноразовых задач настройки, приложение находится в неактивном состоянии.
* **application:didFinishLaunchingWithOptions** -  метод вызывается перед отображением окна приложения, позволяет выполнить код, прежде чем показать приложение пользователю.

Переход на передний план (Foreground):

* **applicationDidBecomeActive** - дает знать, что приложение становится foreground приложением (приложение перешло из неактивного состояние в Active или, в случае, если пользователь игнорирует прерывание (входящий вызов, SMS), которое отправило приложение в неактивное состояние). Метода используется для перезапуска задач
* **applicationWillEnterForegrou**nd - дает знать, что приложение перемещено из фона обратно в foreground, но то, что оно еще не активно (переход из Background в состояние Foreground). Метод applicationDidBecomeActive вызывается вскоре после завершения выполнения данного метода

Переход на задний план (Background):

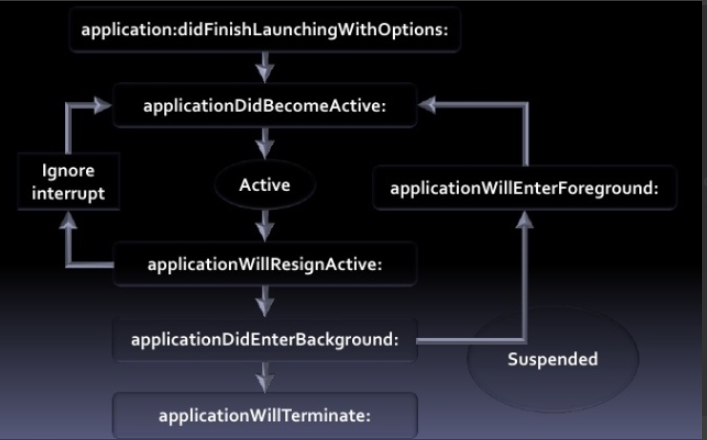
* ***applicationDidEnterBackground****- дает знать, что приложение запущено в фоне и может быть выключено в любое время (переход в состояние Background).*

*Переход в неактивное состояние:*

* **applicationWillResignActive** - дает знать, что приложение уходит из состояния foreground (переключение на другое приложение, входящий вызов, нажатие кнопки «Home»).

Окончание, прекращение работы:

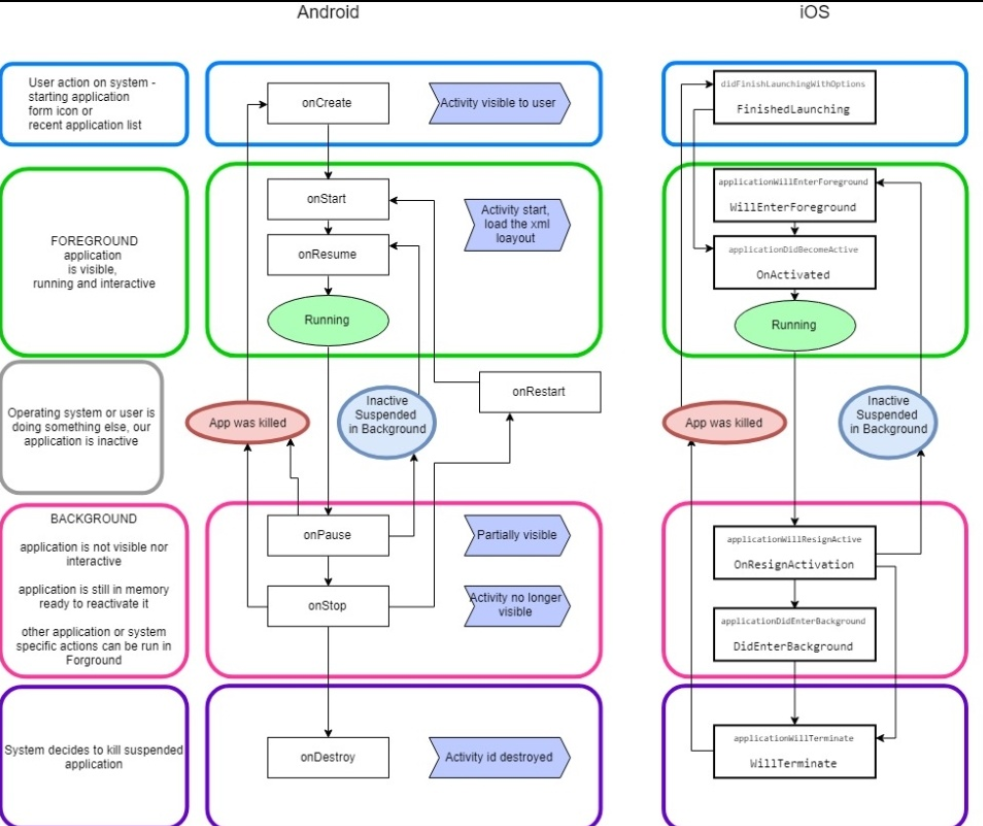
* **applicationWillTerminate** - дает знать, что приложение вот-вот будет выключено (пользователь закрывает приложение). Этот метод может быть вызван в ситуациях, когда приложение работает в фоновом режиме (не приостановлено), и система должна прекратить его по какой-либо причине
  + система не вызывает этот метод при перезагрузке устройства.



Необходимо адаптировать приложения к переходам состояний, чтобы обеспечить лучший пользовательский интерфейс. Понимание основ состояний и переходов позволяет разрабатывать, а затем тестировать более надежные приложения.

### Android vs iOS жизненный цикл приложения

ЖЦ Android и iOS приложений очень похожи. Как видно на схеме ниже сallback-методам активности в Android можно привести в соотвествие методы перехода между состояниями в iOS:



Есть одно отличие, которое можно заметить на схеме. Разница в том, что iOS  пропускает WillEnterForeground событие при "холодном" запуске приложения (из закрытого состояния).