Проверьте использование энергии с помощью Energy Profiler

Energy Profiler поможет вам найти, где ваше приложение потребляет больше энергии, чем необходимо.

Профилировщик энергии контролирует использование процессора, сетевого радио и датчика GPS и отображает, сколько энергии использует каждый из этих компонентов. Профилировщик энергии также показывает наличие системных событий (блокировок пробуждения, аварийных сигналов, заданий и запросов местоположения), которые могут повлиять на потребление энергии.

Профилировщик энергии напрямую не измеряет потребление энергии. Скорее, он использует модель, которая оценивает потребление энергии для каждого ресурса на устройстве.

Обзор профиля энергии

Energy Profiler отображается в виде строки в окне **Profiler** при запуске приложения на подключенном устройстве или эмуляторе Android под управлением Android 8.0 (API 26) или выше.

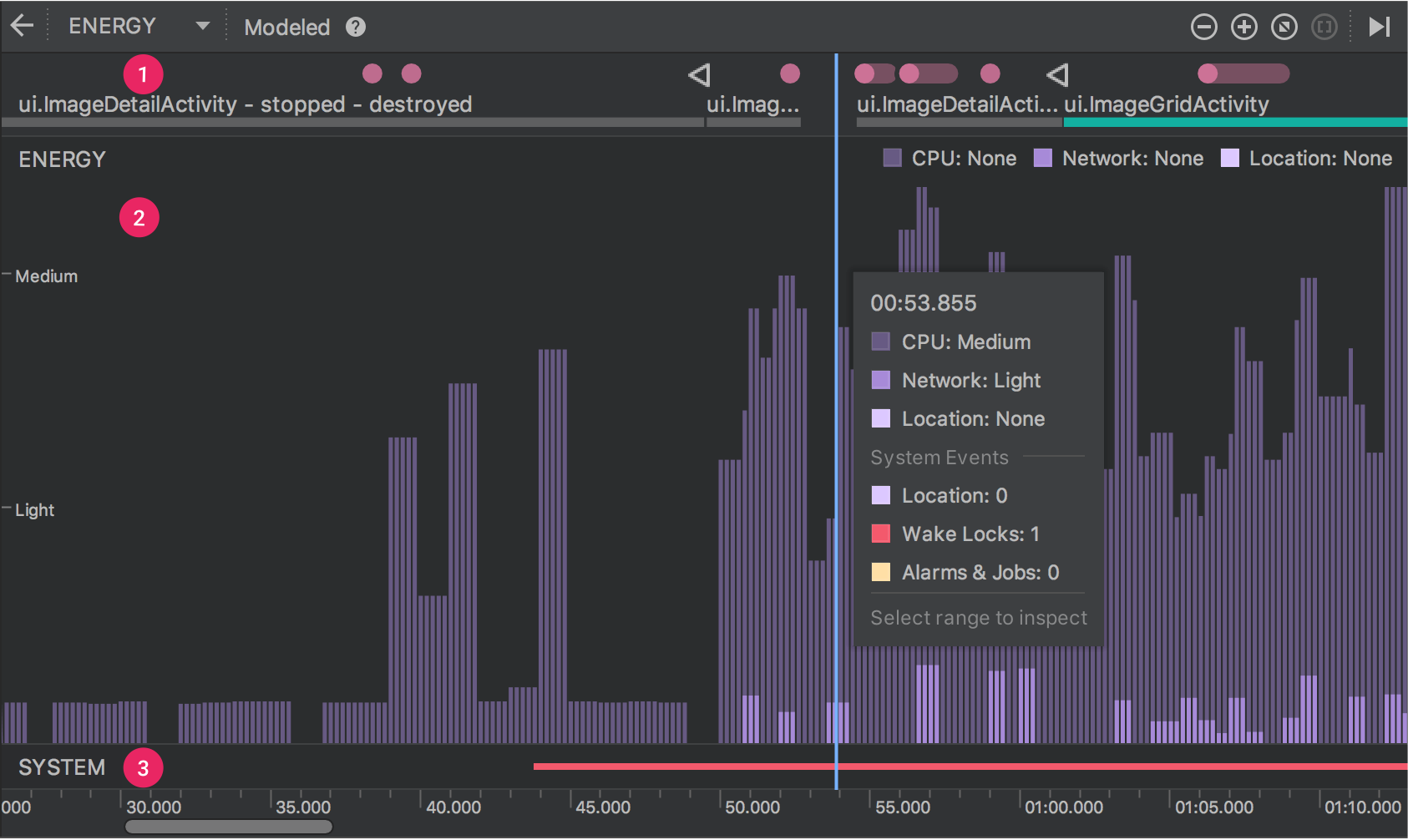
Чтобы открыть Профилировщик энергии, выполните следующие действия:

1. Выберите **View> Tool Windows> Profiler** или щелкните **Profile** https://developer.android.com/studio/images/buttons/toolbar-android-profiler.png на панели инструментов.

Если появится запрос в диалоговом окне « **Выбор цели развертывания** », выберите устройство, на котором будет развернуто приложение для профилирования. Если вы подключили устройство через USB, но не видите его в списке, убедитесь, что вы [включили отладку по USB](https://developer.android.com/studio/debug/dev-options.html#enable) .

1. Щелкните в любом месте на шкале времени **Энергии,** чтобы открыть Профилировщик энергии.

Когда вы открываете Energy Profiler, он сразу начинает отображать предполагаемое потребление энергии вашим приложением. Вы должны увидеть нечто похожее на рисунок 1.



**Рисунок 1.** Временные шкалы в профиле энергии.

Как показано на рисунке 1, представление по умолчанию для Energy Profiler включает в себя следующие временные шкалы:

1. **Временная шкала событий:** показывает действия в вашем приложении, когда они переходят через разные состояния в своем жизненном цикле. Эта временная шкала также указывает на взаимодействие пользователя с устройством, включая события поворота экрана.
2. **График энергопотребления:** показывает приблизительное потребление энергии вашего приложения.
3. **Системная шкала времени:** показывает системные события, которые могут повлиять на потребление энергии.

Чтобы увидеть разбивку использования энергии ресурсами ЦП, сети и местоположения (GPS), а также соответствующими системными событиями, наведите указатель мыши на полоску на временной шкале **энергии** .

Проверять системные события: пробуждения, блокировки, задания и тревоги

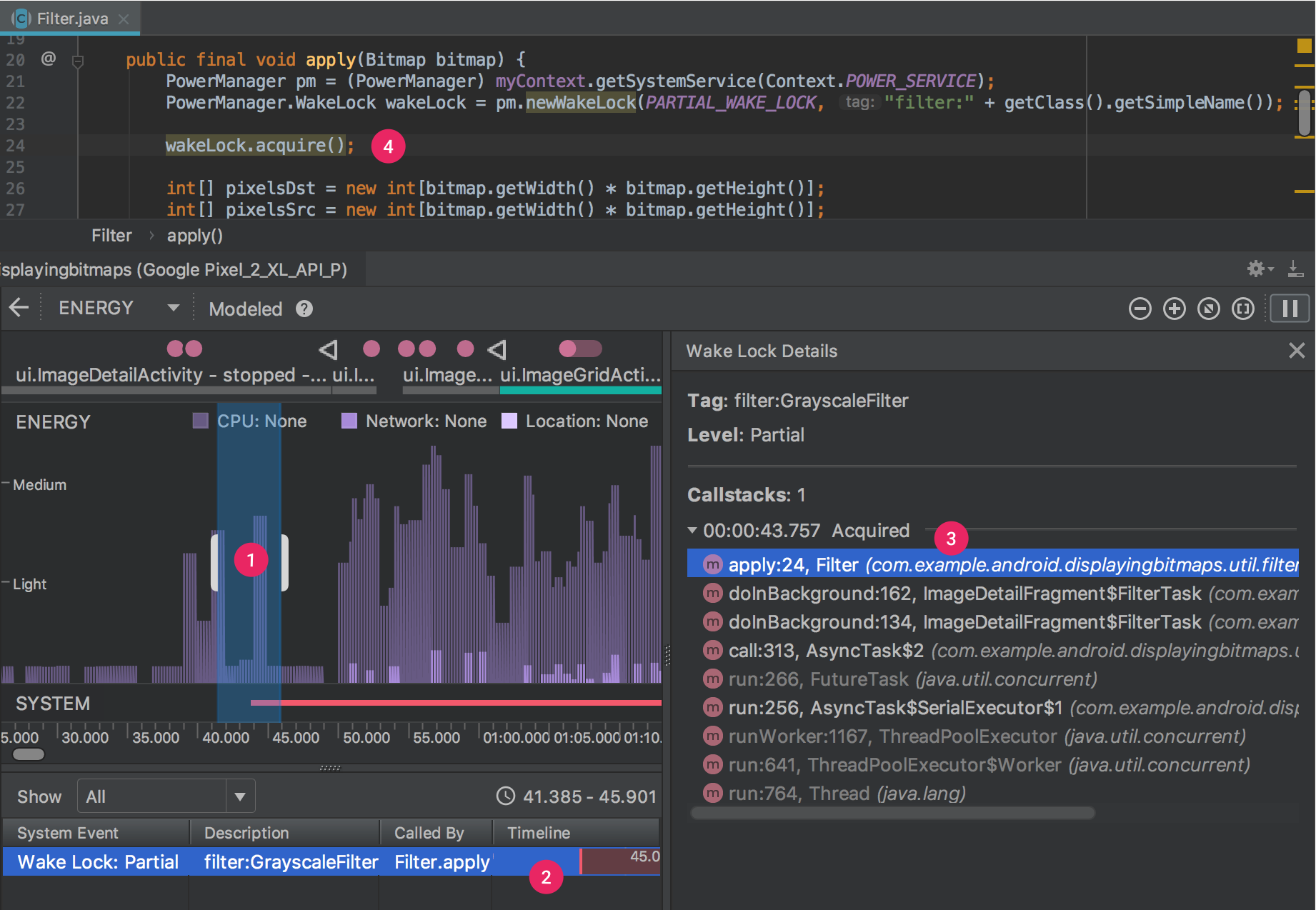
Вы можете использовать Energy Profiler, чтобы найти системные события, которые могут повлиять на потребление энергии, включая пробуждения от блокировки, задания и аварийные сигналы:

* [Замок бодрствования](https://developer.android.com/training/scheduling/wakelock) является механизмом для поддержания процессора или экрана, когда устройство будет в противном случае пойти спать. Например, приложение, которое воспроизводит видео, может использовать блокировку пробуждения, чтобы держать экран включенным, когда пользователь не взаимодействует с устройством. Запрос блокировки пробуждения не является энергоемким действием, но неспособность разблокировать пробуждение может привести к тому, что экран или процессор останутся включенными дольше, чем это необходимо, что может быстро разрядить аккумулятор. Для получения дополнительной информации см. Руководство по [работе с пробужденными замками](https://developer.android.com/topic/performance/vitals/wakelock) .
* Вы можете использовать [сигналы тревоги](https://developer.android.com/training/scheduling/alarms) для регулярного запуска фоновых задач вне контекста вашего приложения. При срабатывании будильника устройство может проснуться и запустить энергоемкий код. Для получения дополнительной информации см. Руководство по [работе с аварийными сигналами](https://developer.android.com/topic/performance/vitals/wakeup) .
* Вы можете использовать [задания](https://developer.android.com/reference/android/app/job/JobScheduler.html) для выполнения действий в определенных условиях, например, когда сеть станет доступной. Вы создаете рабочие места JobBuilderи планируете их JobScheduler. Во многих случаях рекомендуемый подход заключается в использовании заданий, JobSchedulerа не аварийных или пробужденных замков.
* Запросы местоположения используют датчик GPS, который может потреблять значительное количество энергии. Для получения информации о том, как сделать энергоэффективные запросы местоположения, посмотрите [Оптимизацию местоположения для батареи](https://developer.android.com/guide/topics/location/battery) .

Energy Profiler позволяет легко найти, где ваше приложение использует каждую из этих функций, чтобы вы могли принимать обоснованные решения о том, как каждая из них используется.

Профилировщик энергии отображает полосу с цветовой кодировкой на временной шкале **системы под** временной шкалой **энергии** для диапазона времени, когда системное событие активно. Блокировки от пробуждения обозначены красной полосой, задания и тревоги обозначены желтой полосой, а события местоположения - светло-фиолетовой полосой.

На рисунке 2 показан Energy Profiler и редактор кода, открытый для исходного кода, ответственного за невыпущенную блокировку пробуждения.



**Рисунок 2.** Использование профилировщика энергии для поиска пробуждения.

1. Чтобы открыть панель « **Системные события»** и показать подробные сведения о таких событиях, как блокировки после пробуждения, выберите временной диапазон на временной шкале **энергии** .
2. Чтобы открыть панель сведений о **Wake Lock** и показать детали для конкретной **Wake Lock** , выберите **Wake Lock на** панели **System Event** .
3. Чтобы открыть редактор кода и перейти к исходному коду для блокировки пробуждения, дважды щелкните запись для вызывающего метода в верхней части стека вызовов в области **сведений о блокировке пробуждения** .
4. Вызов для получения блокировки пробуждения выделяется в редакторе исходного кода.

Инструкции для отображения подробностей для других системных событий, по сути, такие же, как и для пробужденных блокировок, с информацией, относящейся к каждому виду событий, в своей области сведений. Например, на панели « **Сведения** о задании» отображаются стеки вызовов для частей кода, где задание запланировано и где задание завершено.