Используем Hierarchy Viewer для оптимизации Android приложения

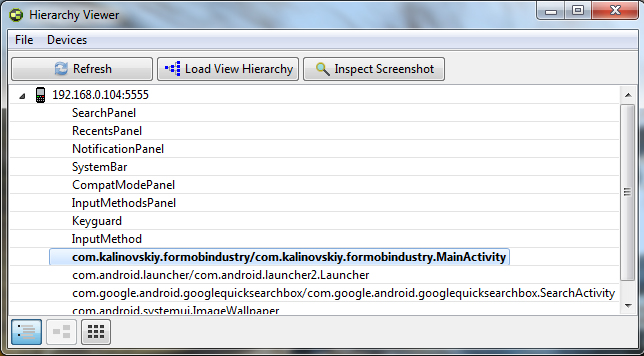
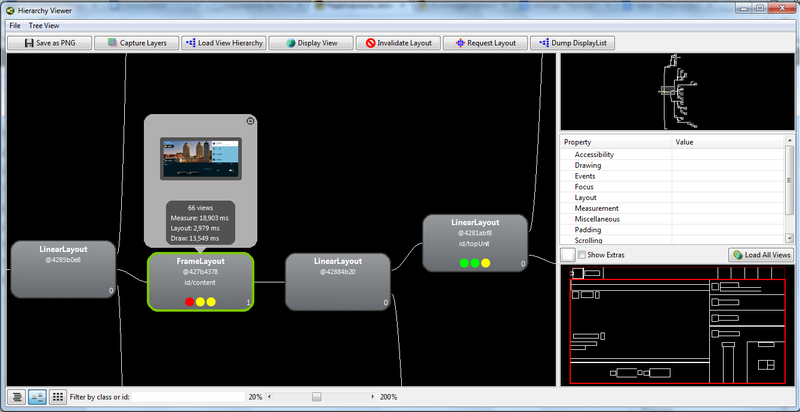
**Hierarchy Viewer** — полезная утилита, предназначенная для оптимизации UI Android приложения, а также для визуализации и проверки компонентов, которые его составляют. Для запуска Hierarchy Viewer нужно прописать:  
**hierarchyviewer**  
в командной строке из папки *android-sdk\platform-tools*.  
В IntelliJ IDEA можно осуществить запуск прямо из среды (*Tools — Android — Hierarchy Viewer*).

Для удобства запуска Hierarchy Viewer из командной строки нужно выполнить следующие действия (актуально для Windows):

1. Правый клик на Мой компьютер — Свойства — Дополнительные параметры системы — Переменные среды;
2. В системных переменных нужно найти переменную *Path*, нажать «Изменить», и в конце строки добавить: **;C:\[ваш\_путь\_к\_android-sdk]\platform-tools**
3. Теперь можно выполнять команду запуска прямо через «Пуск-Выполнить».

После того, как Hierarchy Viewer успешно запущен, нужно подключить Android-девайс (если это не было сделано раньше) и на экране появятся запущенные на устройстве процессы.

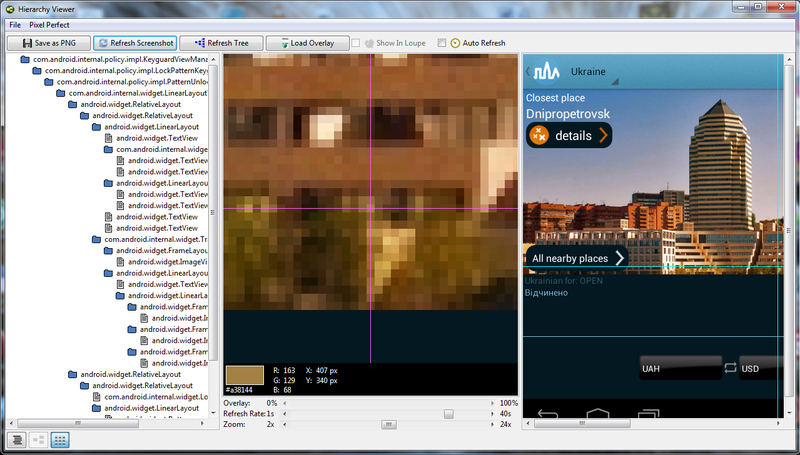
*Hierarchy Viewer работает только с рутоваными устройствами, либо эмулятором.*

  
  
После выбора процесса, нуждающегося в оптимизации перейдем в окно просмотра иерархии — *Load View Hierarchy*. Все Layouts и Views активного Activity отображены в виде дерева (слева направо).  
  
  
  
Элементы имеют разноцветные индикаторы, которые указывают на скорость рендеринга данного view/layout по отношению к другим объектам в дереве. Для получения подробных данных нужно нажать на элемент. Цвета указывают на следующие относительные показатели:

* Зеленый цвет означает, что элемент среди 50% наиболее быстро загружаемых view в Activity.
* Желтый цвет означает, что элемент — среди 50% наиболее медленно загружаемых.
* Для загрузки элементов с красным индикатором требуется больше всего времени в дереве.

В случае избытка желтых и красных индикаторов, имеет смысл внести изменения в структуру Activity, уменьшить уровень вложенности, избавиться от ненужных View, использовать другой формат изображений etc.  
  
На панели инструментов доступны следующие действия:

1. *Save as PNG* сохранит все дерево в png-файл;
2. *Capture Layers* сохранит скриншот экрана в формате psd, при этом каждый view будет расположен на отдельном слое;
3. *Load View Hierarchy* заново перерисует дерево иерархии;
4. *Display View* откроет скриншот экрана в новом окне;
5. *Invalidate Layout* отметит view как такой, который не будет перерисован;
6. *Request Layout* перерисует view и все его наследующие;
7. *Dump DisplayList* сделает дамп дерева для загрузки его через меню Tree View — Load View Hierarchy.

Hierarchy Viewer также может работать со скриншотом Activity приложения. Вернемся в окно процессов, нажав на первую из трех навигационных кнопок в левом нижнем углу, после чего предстоит выбрать процесс и нажать *Inspect Screehshot*. На скриншоте можно проверить каждый пиксель, его цвет в RGB и координаты.  
  
  
  
На панели инструментов также есть функции:

1. *Save as PNG* сохранит скриншот в png-файл;
2. *Refresh Screenshot* обновит его;
3. *Refresh Tree* обновит дерево иерархии, которое отображено в левой части;
4. *Load Overlay* позволит наложить на скришнот полупрозрачное изображение, например, для сравнения мокапа с реальным приложением.

На этом функционал Hierarchy Viewer заканчивается. Как видно, утилита достаточно проста в использовании, с ее помощью вы сможете улучшить показатели скорости загрузки вашего приложения.