Android Game Development

Using OpenGL ES 2.0

Android Game Development

- 2D games easier to build than I imagined
- I’m not a professional game developer
- https://github.com/beamtech/OpenGLSampleGame.git

{ “hip”, “hip” } 

Boulders & Band-Aids

- ⭐ Casual 2D game
- ⭐ Rolled our own simple game engine on top of OpenGL
- ⭐ Game play is a reward for brushing your teeth :)
- Part of the **Beam Dental** app

Why did we choose OpenGL?

- OpenGL 2.0 ES supported on almost all devices
- Our team already comfortable with it
- Mature technology
- 2D game — relatively simple

Cons of OpenGL

- High barrier to entry
- Low level graphics system, not a game engine
- No built-in physics engine, collision detection
- Other options
  - Unity, libgdx, AndEngine, and many more…

Boulders & Band-Aids Architecture

- ❤️ OpenGL view and renderer
- ❤️ Drawable objects (Sprites)
- ❤️ Engine and levels
- ❤️ Player input
- ❤️ Sounds

GLSurfaceView

- A view like any other Android view
  - But allows you to draw with OpenGL
- Subclass it
  - Set the OpenGL version to use
  - Set your renderer
  - Set rendermode (continuous or not)

GLSurface.Renderer

- onSurfaceCreated()
  - Called when view is created/recreated, good place to initialize/ recreate textures
- onSurfaceChanged()
  - Called when view changed size
- onDrawFrame()
  - Called to draw the current frame
- **Note: These all run on a GLThread!**

Drawable Objects

- Texture Sprites
- Lines (high score line)
- Text
  - Surprisingly hard to draw in OpenGL
  - https://github.com/beamtech/GLText

Engine and levels

- Engine
  - Game loop: update state and redraw everything
  - Keeps game state
  - Also handles player input and sound
- Level management
  - Levels loaded from external JSON file

Player Input

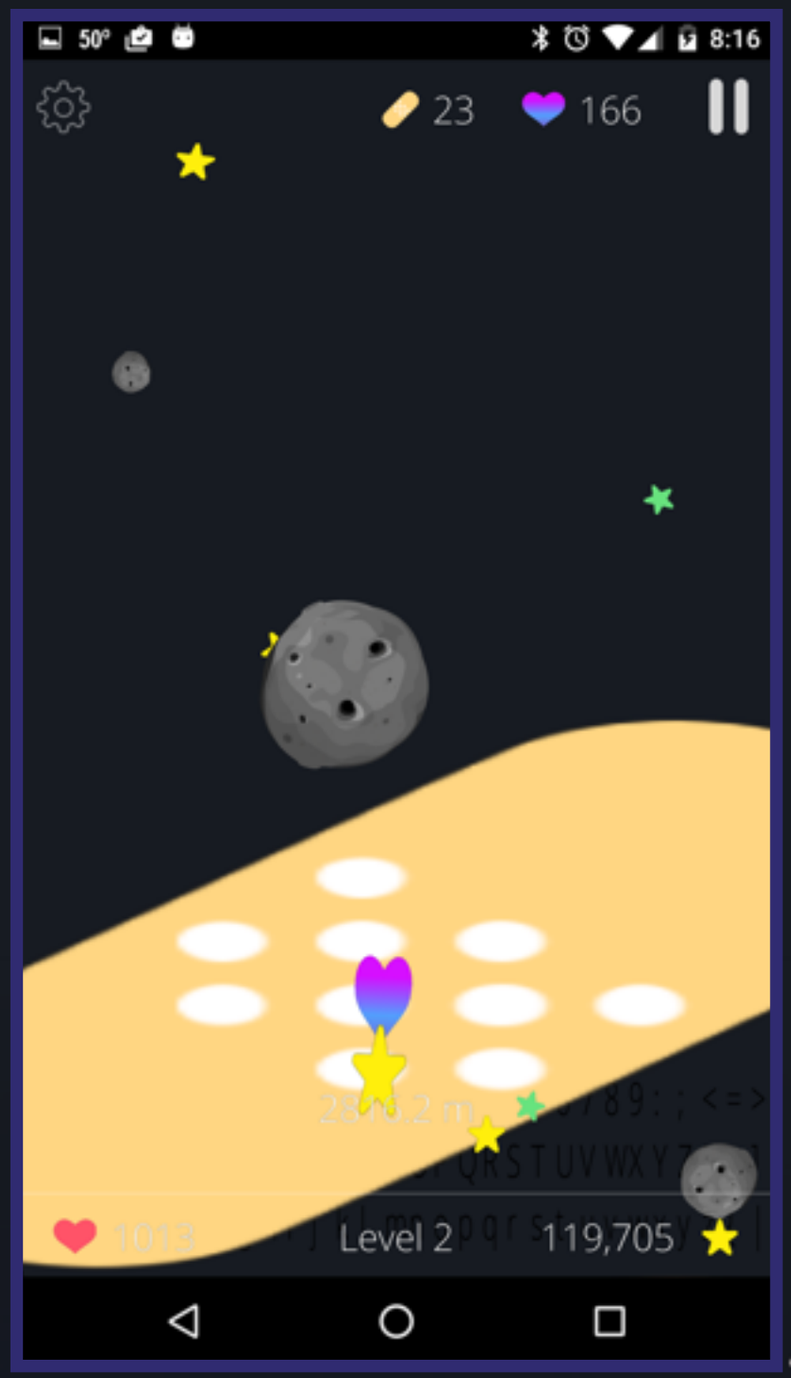
- Tilt phone to move character
- Android sensors
- Accelerometer, Magnometer, Gyroscope

Sounds

- Use Android's SoundPool API for sound effects
  - Lower latency
  - Easier API
  - Better looping
- Use MediaPlayer for longer sounds or music

Gotchas

- Running on GLThread
- Reload OpenGL resources/ textures when app returns from background



Performance Tips

- Only 16ms! Use Systrace and Traceview
- Avoid Android views w/ alpha over GLSurfaceView
- Pre-allocate memory to avoid GC lags
  - Sprite pooling
- Batch OpenGL draw operations

Resources

- Peter Hewitt
  - peter@hewittsoft.com
  - https://github.com/dweebo
- https://github.com/beamtech/OpenGLSampleGame.git
- http://developer.android.com/guide/topics/graphics/opengl.html
- http://developer.android.com/tools/help/systrace.html
- http://blog.udinic.com/2015/09/15/speed-up-your-app