Projector: Your Remote IDE

Pasha Finkelshteyn, JetBrains

asm0di0

Did you ever complain?

- You need to set up a VPN to access data?
- You need to use RDP to access your instance of IDEA?
- Or even VNC?
- *Long* indexing?
- Huge project compiles forever?

Did you ever want?

- Edit code from tablet/smartphone?
- Help your colleague to fix some issues?
- Show demo which accesses data from your cloud?
- Work from a weak laptop?


PJ

Issues with existing solutions. VPN

1. OpenVPN does not scale
    1. No central identity management
    2. No failover OOTB
2. Proprietary solutions are… proprietary
3. And expensive sometimes
4. IKE is not completely cross-platform

Issues with existing solutions. VNC

1. **Extremely** slow
2. Complex infrastructure (VPN?)

Issues with existing solutions. RDP

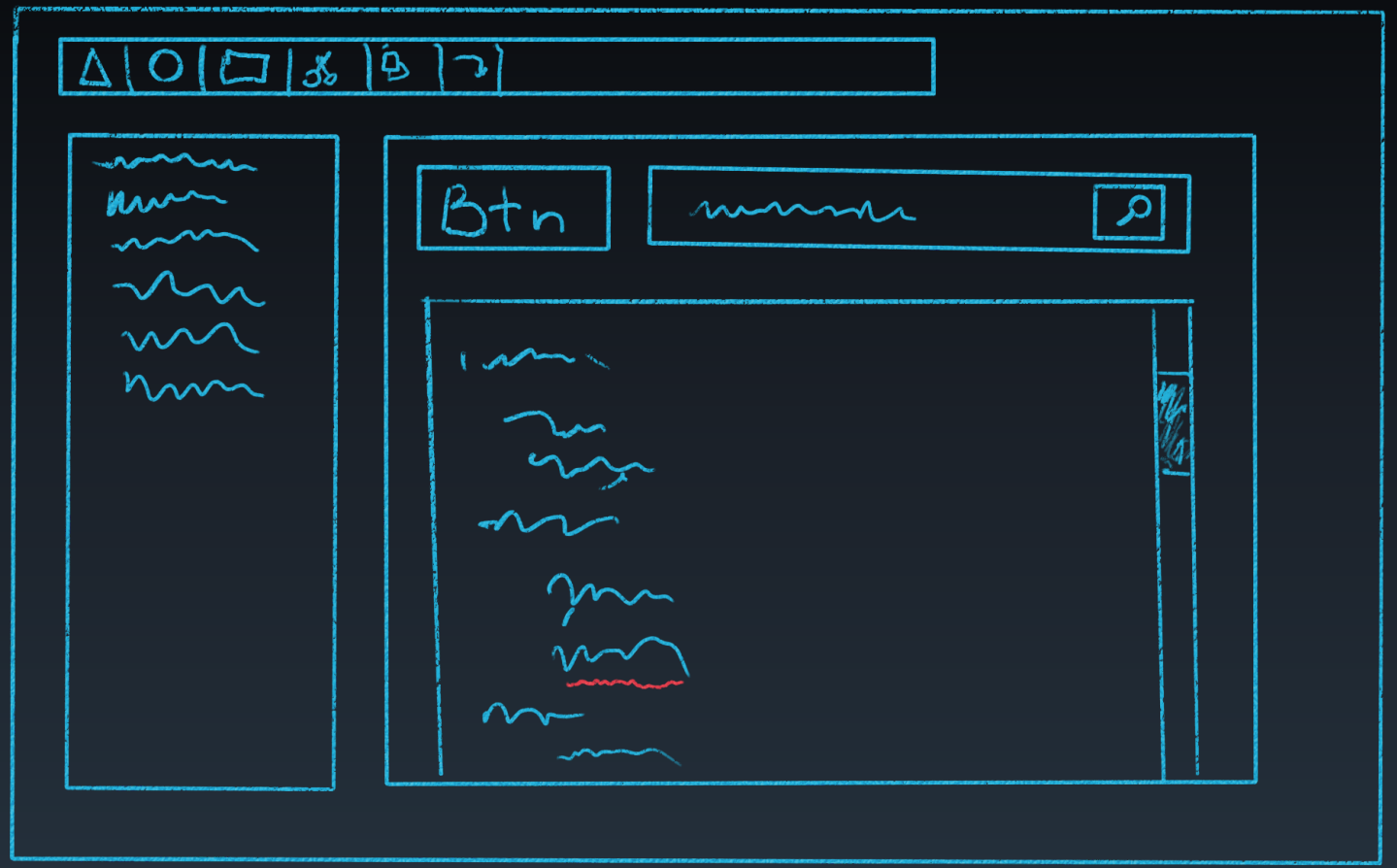
RDP is awesome

1. Single-user
2. Requires special client
3. Still needs complex infrastructure to provide access

Stop this

Let's dig!

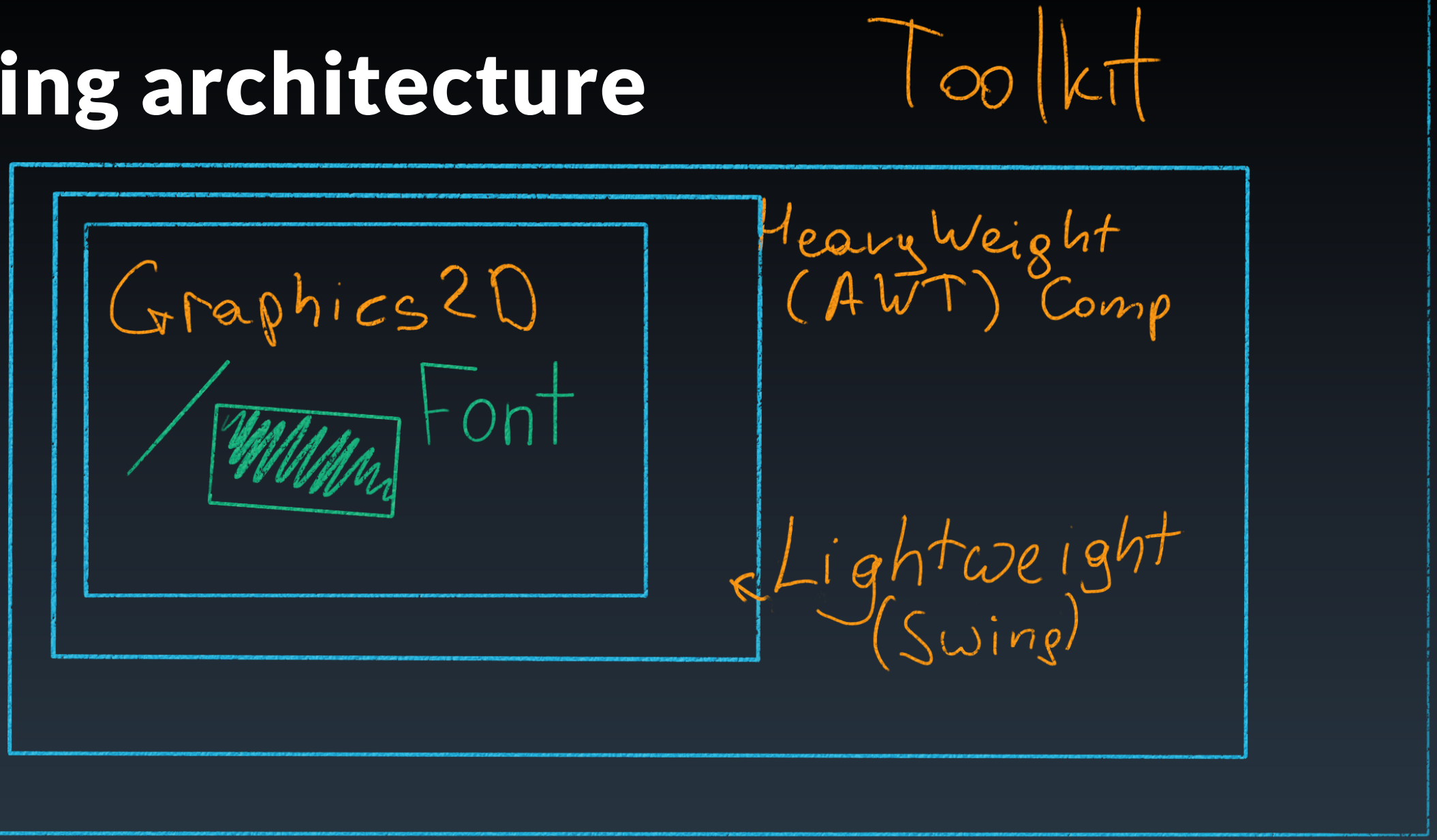
Typical Swing interface



And in reality?

- Lines
- Simple shapes
- Strings
- Fonts

Swing architecture



Definitions

Toolkit — abstraction over the whole graphics
Heavyweight components — system-controlled components
Lightweight components — user-controlled components
Graphics2D — drawing controller

Types of drawing surfaces

- Visible
- Invisible
  Part of the editor which you don't see is separate and is drawn on an invisible surface

Each window has its own surface(s)

Heavyweight components

There is a corresponding `Peer` class for each heavyweight component.

```
interface ComponentPeer {
    void paint(Graphics g);
    void setBounds(int x, int y, int width, int height, int op);
    void handleEvent(AWTEvent e);
    void setBackground(Color c);
    void setFont(Font f);
    // etc.
```

Responds for delegation to Graphics and setting several params

Heavyweight components

We have implementations of peers only for several heavyweight components.

Graphics2D

Anything may contain Graphics2D

Responds for drawing:

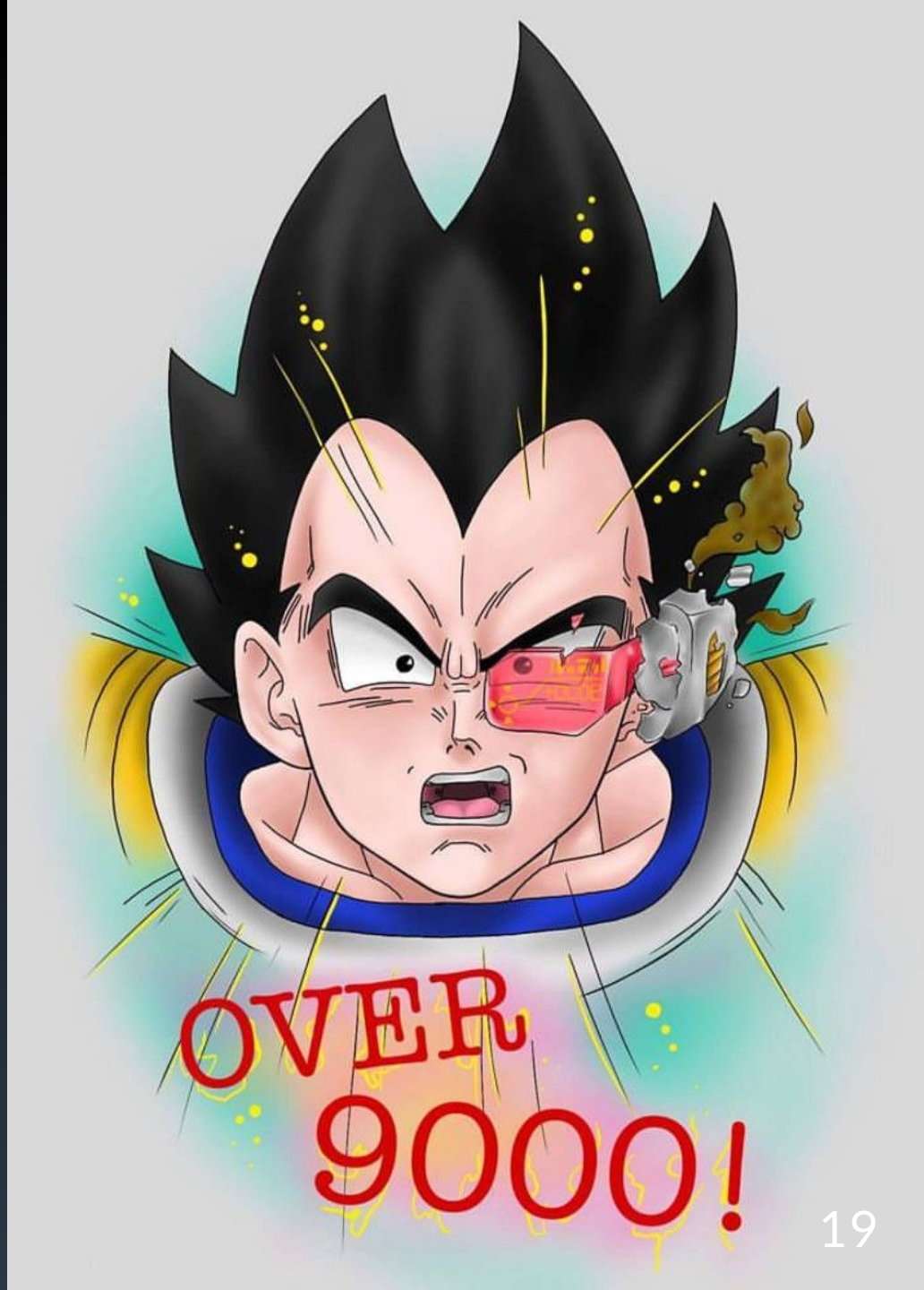
```
abstract class Graphics2D {
    public abstract void drawImage(BufferedImage img,
                                   BufferedImageOp op,
                                   int x, int y);
    public abstract void drawString(String str, int x, int y);
    public abstract void fill(Shape s);
    // etc.
```

How the hell will it work in a browser?

How the hell will it work in a browser?

Initial version just sent the whole screen to the client.

It had whopping performance of ≈5 fps 😯



How the hell will it work in a browser?

Welcome DrawEventQueue

Queue, that will transform any action into event, based on

```
ConcurrentLinkedQueue<List<ServerWindowEvent>>()
```

DrawEventQueue

Each component hash its own queue (for now)

```
private inline fun paintArea(
    crossinline command: DrawEventQueue.CommandBuilder.() -> Unit
) {
    drawEventQueue
        .buildCommand()
        .setClip(identitySpaceClip = identitySpaceClip)
        .setTransform(transform.toList())
        .setComposite(backingComposite)
        .command()
}
```

Images

There are at least 5 times of images in AWT 😱

`VolatileImage`:

- is used to display off-screen surfaces
- is drawn by primitives
- is being sent as commands.

Images

`BufferedImage`:

- is being sent as bitmap
- we have sophisticated reflection-based heuristics to determine if it is changed
- may be optimized in future

Images

`MultiresolutionImage`:

- used for drawing icons
- supported only if it has the single resolution

Images

Other types of images are not supported right now.

So we have commands. Now what?

```
while(!stopped){
    sendAllCommands() // sends both images and command
    Thread.sleep(10)
}
```

Every 10ms (roughly) aggregate all commands and send them to the client over WebSocket.

Format

Default format: highly optimized JSON
Supported OOTB formats: Protobuf, JSON
May support anything if the browser supports it.

Format

```
[["e",{"a":["a",{"a":2}],"b":[["h",{"a":["a",{"a":252.0,"b":78.0,
"c":26.0,"d":24.0}]}],["p",{"a":[1.0,0.0,0.0,1.0,21.0,76.0]}],["i",
{"a":["a",{"a":1.0,"b":"a","c":"c","d":10.0,"e":0.0,"f":null}]}],
["s",{"a":["a",{"a":-12828863}]}],["r",{"a":["a",{"a":"a","b":1.0}]}]
,["d",{"a":"b","b":0.0,"c":0.0,"d":478.0,"e":28.0}],["p",{"a":[1.0,0.
0,0.0,1.0,252.0,78.0]}],["l",{"a":["a",{"a":1024,"b":-172147019}],
"b":["b",{"a":5,"b":4,"c":16,"d":16,"e":null}]}]]}]]
```

Everything is determined in runtime — types, fields, etc.
For example `"e"` at the very beginning is type!

Backend

- Ktor?
- Tomcat?
- Undertow?
- Netty?
- Jetty?


nope.

GPL is not always a good thing

Popular products are not compatible with it, and we HAVE TO publish source code under GPL license.

So...

TooTallNate/Java-WebSocket

And a bunch of crutches to add HTTP support there.

Clients

A client should implement ≈30 API methods of drawing

The good part of Kotlin: we can share some logic between client and server (and we do).

`kotlilnx.serialisazation` *may* do things easier (but deserialization is sooo slow).

Non-read-only client should be able to send mouse and keyboard events, request for resources and so on.

In browser

CompareVerifier – ComparableVerifier.java [CompareVerifier.main] — Mozilla Firefox

127.0.0.1:9878

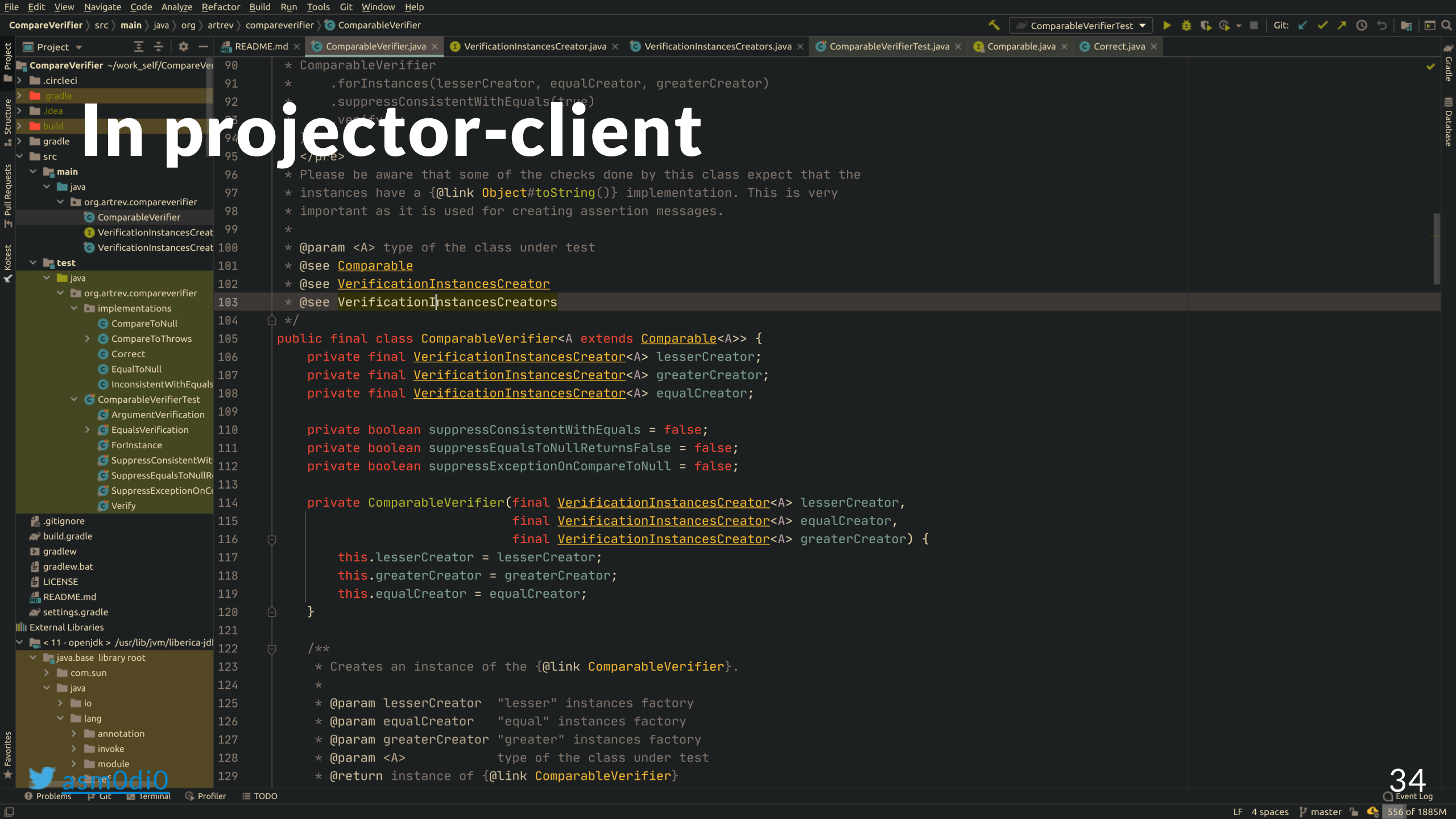
File Edit View Navigate Code Analyze Refactor Build Run Tools Git Window Help

```
 * @param lesserCreator  "lesser" instances factory
 * @param equalCreator   "equal" instances factory
 * @param greaterCreator "greater" instances factory
 * @param <A>            type of the class under test
 * @return instance of {@link ComparableVerifier}
 */
public static <A extends Comparable<A>> ComparableVerifier<A> forInstances(
        final VerificationInstancesCreator<A> lesserCreator,
        final VerificationInstancesCreator<A> equalCreator,
        final VerificationInstancesCreator<A> greaterCreator) {

    return new ComparableVerifier<A>(
            lesserCreator, equalCreator, greaterCreator
    );
}

/**
 * Causes that the (a.compareTo(b)==0) == (a.equals(b)) won't be verified.
 * <p>
 * According to the {@link Comparable} documentation it is strongly
 * recommended, but not strictly required that aforementioned rule is obeyed.
 * <p>
 * It is advised that any class that implements the {@link Comparable}
 * interface and violates this particular rule should clearly indicate this
 * fact.
 *
 * @param suppressCheck true if the rule should be suppressed.
 * @return instance of {@link ComparableVerifier}
 */
public ComparableVerifier<A> suppressConsistentWithEquals(final boolean suppressCheck) {
    suppressConsistentWithEquals = suppressCheck;
    return this;
}

/**
```


In projector-client
asm0di0
34

Desktop client

- Electron-based
- (Almost) all browser hotkeys are disables
- May me expanded to full screen

Installation: GNU/Linux

```
pipx install projector-installer
```

```
❯ projector run
           1. DataGrip
           2. IntelliJ
           3. IntelliJ_IDEA_Toolbox_ch-0
Choose a configuration number or 0 to exit: [0-3]: 1
Configuration name: DataGrip
To access IDE, open in browser
        http://localhost:9876/?token=qwertyuiop%21%40%23%24%25%5E%26%2A%28%29
        http://127.0.0.1:9876/?token=qwertyuiop%21%40%23%24%25%5E%26%2A%28%29
        http://192.168.1.111:9876/?token=qwertyuiop%21%40%23%24%25%5E%26%2A%28%29

To see Projector logs in realtime run
        tail -f "/home/finkel/.projector/configs/DataGrip/projector.log"

Exit IDE or press Ctrl+C to stop Projector.
```

Installation: Docker

JetBrains/projector-docker

```
docker pull \
    registry.jetbrains.team/p/prj/containers/projector-idea-u
docker run --rm -p 8887:8887 -it \
    registry.jetbrains.team/p/prj/containers/projector-idea-u
```

Useful scripts

- Build a container with IDE of your choice

  `build-container.sh [containerName [ideDownloadUrl]]`

- Run Stateless container

  `run-container.sh [containerName]`

- Run container with mounted directory to preserve state

  `run-container-mounted.sh [containerName]`

How does it compare to CWM?

- It's not a tool for collaborative development (for now)
- Allows to display any IDE component
- Very comfortable to help/comment on somebody's issues

Place for improvement

- Visual glitches
- Support for (more) AWT components
- Improve HiDPI support
- Speculative typing (?)
- iOS (Android?) native clients (Flutter?)
- Queue optimization
- **You** name it

What did we learn?

- UI operations can be buffered
- There are not so many REAL UI operations
- Crossplatform drawing is achievable
- Parts of code may really be reused between server and client

projector.jetbrains.com

Thank you

asm0di0

asm0di0