



Increasing subjective testability with Fiddler

Rasmus Koorits
RSI

@rkoorits



Get Fiddler 4

- Get Fiddler from <https://www.telerik.com/download>



Structure of the workshop

- Explain what I mean by “Increasing subjective testability”
- Cover some basic concepts
- Explain how fiddler works
- Switching between theory segments and practice sessions
- Ask questions at any point!
- Coffee break somewhere in the middle
- Share your testability issues and we can solve them together



On Testability

- Two cool models!

Epistemic Testability

How narrow is the gap between what we know and what we need to know about the status of the product under test?

Example: "Life-critical software is harder to test."

Project-Related Testability

Testability influenced by changing the conditions under which we test.

Example: "When the developers discuss upcoming changes with me, I can target my testing better."

Value-Related Testability

Testability influenced by changing the quality standard or our knowledge of it.

Example: "Testing is easier now that I work with the people who are going to use this system."

Practical Testability

Subjective Testability

Testability influenced by changing the tester or the test process.

Example: "Ever since I learned Javascript, testing this web application has been easier."

Intrinsic Testability

Testability influenced by changing the product itself.

Example: "The new function-level log file records everything I do when I test."



On Testability

- Visibility
 - How can we see what is going on inside the system
 - How can we know if what we did had the expected result
 - ... and did not have unexpected results
- Controllability
 - How can we alter the state of the system
 - Or the data in the system



On Testability

- Proxy servers can help with both Visibility and Controllability when testing in a client/server environment!
- ... but we need some practice in order to realize when to use them.



Pattern recognition for tool selection

- The ability to recognize a familiar problem
- And picking the appropriate tool (or technique)



Pattern recognition for tool selection

- A loose screw
- Leaking pipe
- Testing the maximum size of a large text field
- Testing a text editor

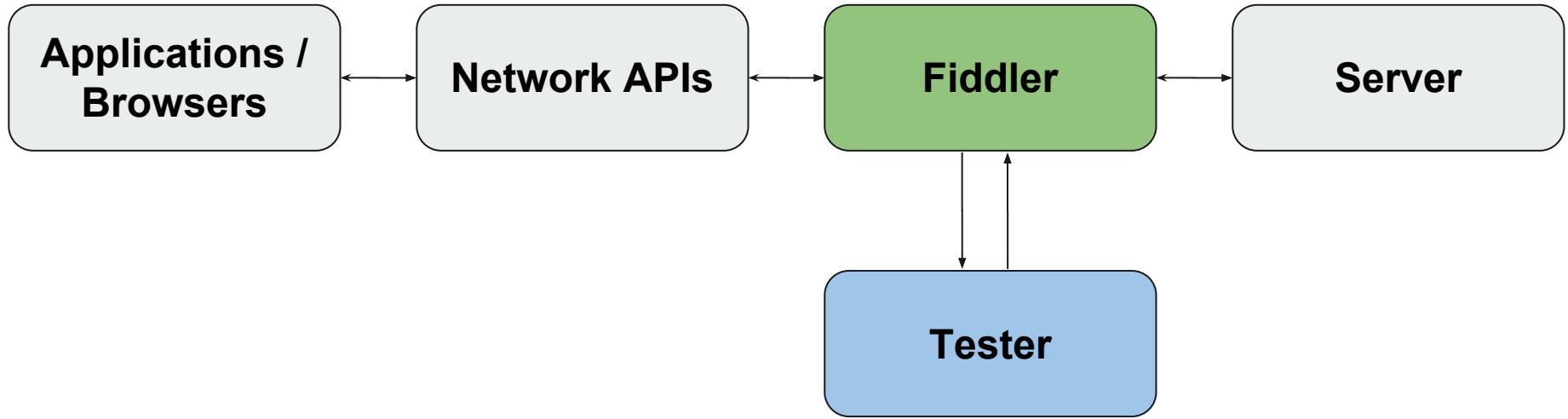


Goals of this workshop

- Learn about problem patterns that proxy tools can help with
- Gain hands-on experience to facilitate the above
- Try to solve some real life testability problems as a group
- You start using proxy servers in your daily work



What is a Proxy?





Any questions so far?



HTTP(s) Primer - Request/Response paradigm

- Clients make a **Request**
- Servers return a **Response**
- ... thus forming a request/response pair
- Fiddler calls this a **Session**
- (Except for websocket connections, which we will cover later)

Web apps go to a great lengths to hide the basic paradigm!

(because it just looks better that way - consider chatrooms or email clients)

HTTP(s) Primer - sample request

```
POST http://webdbg.com/sandbox/shop/checkout.asp HTTP/1.1
Accept: text/html, application/xhtml+xml, image/jxr, */*
Referer: http://webdbg.com/sandbox/shop/
Accept-Language: en-US,en;q=0.7,et;q=0.3
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
Content-Length: 77
Host: webdbg.com
Connection: Keep-Alive
Pragma: no-cache
Cookie: ASPSESSIONIDSQBACTQ=KNCHFJGJCGEIPDBIDMCDEDCCM|
affiliate=&Cost=1095.00&Item=Acer+Convertible&lbQuantity=3&btnOrder=Check+out
```

The diagram illustrates the structure of an HTTP request. Green arrows point from labels to specific parts of the request:

- Request line**: Points to the first line of the request: `POST http://webdbg.com/sandbox/shop/checkout.asp HTTP/1.1`
- Headers**: Points to the block of header lines starting with `Accept:` and ending with `Cookie:`.
- Empty line**: Points to the blank line that separates the headers from the body.
- Body**: Points to the request body, which is a URL-encoded string: `affiliate=&Cost=1095.00&Item=Acer+Convertible&lbQuantity=3&btnOrder=Check+out`



HTTP(s) Primer - Request Methods

- GET
- POST
- PUT
- DELETE
- OPTIONS
- ...

But these are at *least partially* arbitrary

HTTP(s) Primer - sample response

HTTP/1.1 200 OK

Status line

Date: Sat, 26 May 2018 11:18:58 GMT

Server: Apache/2.4.25 (FreeBSD) OpenSSL/1.0.2k mod_fcgid/2.3.9

X-Powered-By: PHP/5.4.45

Expires: Sun, 19 Nov 1978 05:00:00 GMT

Cache-Control: no-cache, must-revalidate

X-Generator: Drupal 7 (http://drupal.org)

Keep-Alive: timeout=5, max=100

Connection: Keep-Alive

Content-Type: text/html; charset=utf-8

Content-Length: 13734

Headers

Empty line

<!DOCTYPE html>

<!--[if lt IE 7]> <html class="no-js lt-ie9 lt-ie8 lt-ie7" lang="en" xml:lang="en"

xmlns:content="http://purl.org/rss/1.0/modules/content/"

xmlns:dc="http://purl.org/dc/terms/"

xmlns:foaf="http://xmlns.com/foaf/0.1/"

...

Body



HTTP(s) Primer - Response codes

- 200 ← OK
- 404 ← Not Found
- 418 ← I'm a Teapot!
- 500 ← Internal Server Error

But these are *sometimes* arbitrary as well!

... But some tools/frameworks/environments make strong assumptions



HTTP(s) Primer - Response Body

- HTML
- JSON
- XML
- .. or anything else



Any questions so far?



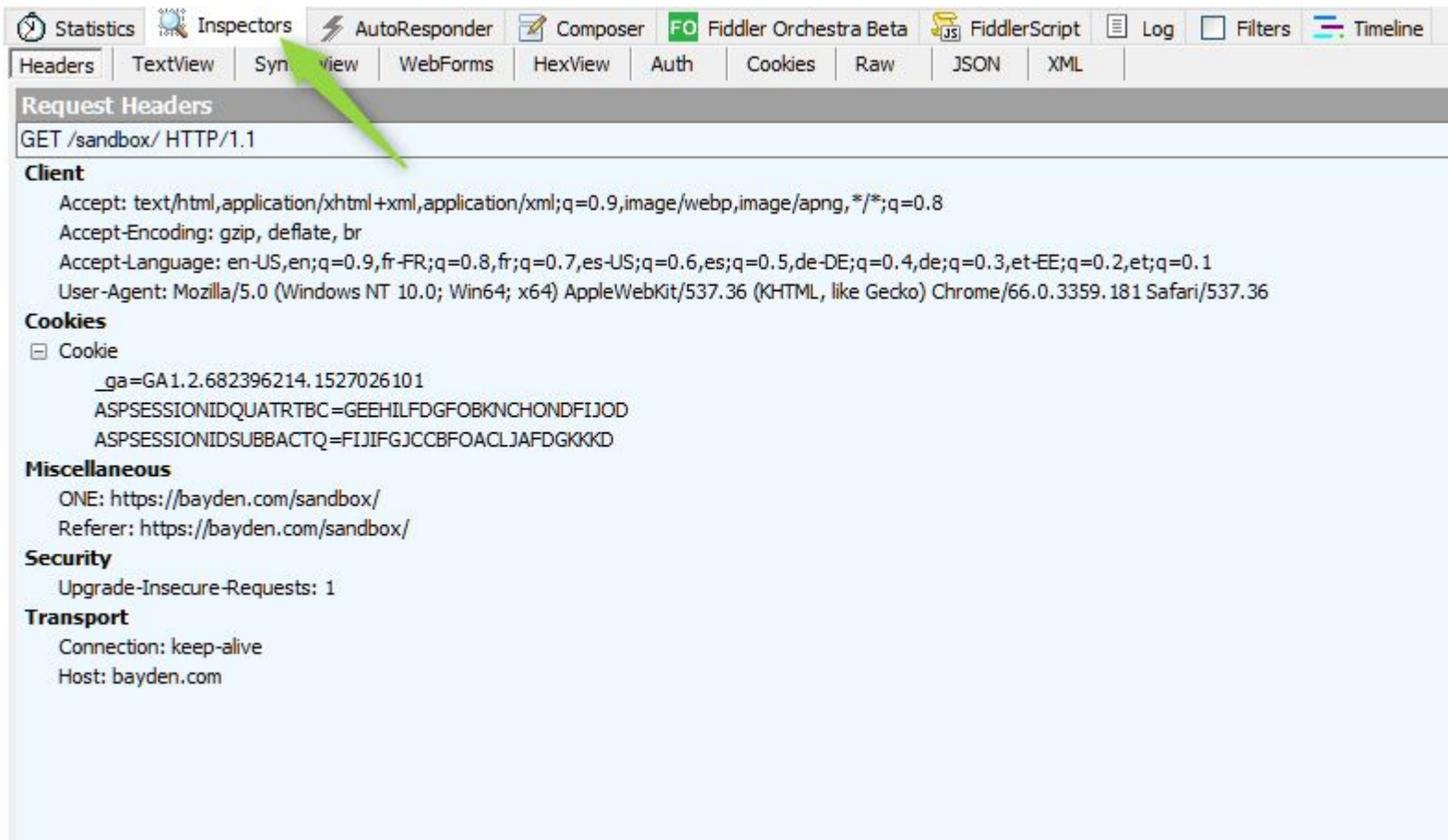
Team up!



Let's run Fiddler!

- As well as Chrome/IE/Edge
- Brace for various warnings at first

Inspectors tab



Statistics Inspectors AutoResponder Composer Fiddler Orchestra Beta FiddlerScript Log Filters Timeline

Headers TextView **Inspector view** WebForms HexView Auth Cookies Raw JSON XML

Request Headers

GET /sandbox/ HTTP/1.1

Client

- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
- Accept-Encoding: gzip, deflate, br
- Accept-Language: en-US,en;q=0.9,fr-FR;q=0.8,fr;q=0.7,es-US;q=0.6,es;q=0.5,de-DE;q=0.4,de;q=0.3,et-EE;q=0.2,et;q=0.1
- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.181 Safari/537.36

Cookies

- Cookie
 - _ga=GA1.2.682396214.1527026101
 - ASPSESSIONIDQUATRBC=GEEHILFDGFOBKNCHONDFIJOD
 - ASPSESSIONIDSUBBACTQ=FIJIFGJCCBFOACLJAFDGGKKD

Miscellaneous

- ONE: <https://bayden.com/sandbox/>
- Referer: <https://bayden.com/sandbox/>

Security

- Upgrade-Insecure-Requests: 1

Transport

- Connection: keep-alive
- Host: bayden.com

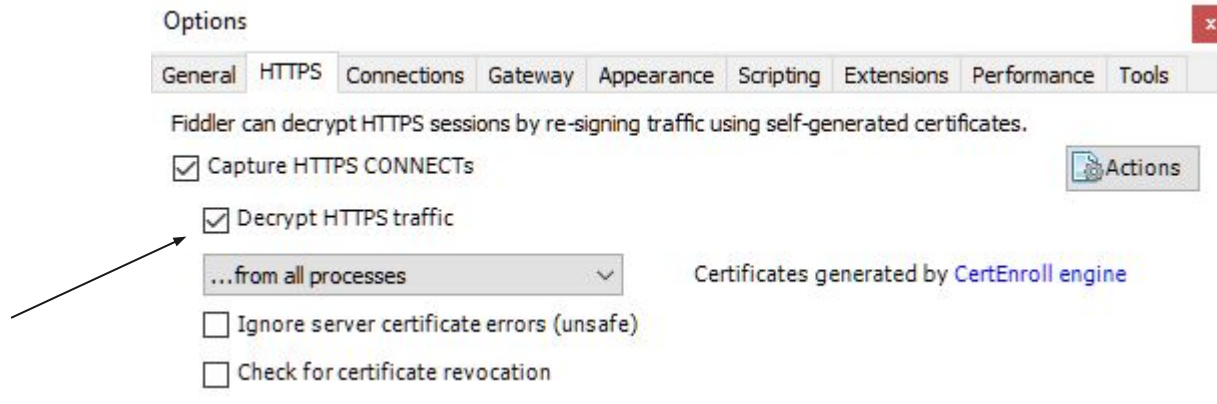


Let's Fiddle stuff!

- **Exercise:** view requests/responses at the Fiddler sandbox - <https://bayden.com/sandbox/>
or at the sandbox created for this workshop:
<http://ntd.codeandtest.org>
- Try out the various Inspectors on different sessions

Enabling HTTPS decryption

File menu: Tools -> Options



See https://en.wikipedia.org/wiki/Man-in-the-middle_attack



Why not just use Chrome Dev Tools?

- Strange Chrome quirks
- Cross browser testing
- Desktop applications
- Phone apps!
- The power of Fiddler!



Filtering out the noise

- Filters tab
- Quick exec
 - bold
- FiddlerScript

Filtering out the noise



Use Filters

Note: Filters on this page are a simple subset of the filtering FiddlerScript offers (click Rules > Customize Rules).

Actions

Hosts

Show only Internet Hosts

Show only the following Hosts

example.com; mysite.com

Client Process

Show only traffic from

Show only Internet Explorer traffic

Hide traffic from Service Host

Request Headers

Show only if URL contains

myApi

Hide if URL contains

Flag requests with headers

Delete request headers

Set request header



Let's Fiddle stuff!

Exercise: Filter to only show requests/responses that have the relevant host names

You can use <https://ntd.codeandtest.org> as a sandbox

Sometimes this is a bad idea, as unexpected calls will go unnoticed!



Breakpoints!

- Tampering Requests before they reach the Server
- Tampering Responses before they are returned to the Client
- Can be enabled from various places:
 - QuickExec: bpu, bpa
 - Filters
 - FiddlerScript
 -



Let's Fiddle stuff!

Exercise 1: Change something in a request

- For example, turn a POST request into a GET request

Exercise 2: Change something in a response

- Turn a 200 OK response into a 500 Error



AutoResponder

- Overwrite a server response
 - With a previously saved response
 - Or a handcrafted one
 - Or even a picture!

AutoResponder tab

Statistics Inspectors **AutoResponder** Composer Fiddler Orchestra Beta FiddlerScript Log Filters Timeline

Fiddler can return previously generated responses instead of using the network. [Help](#)

Enable rules Unmatched requests passthrough Enable Latency

Add Rule Import...

| If request matches... | then respond with... |
|--|----------------------|
| <input checked="" type="checkbox"/> myRule | 200_SimpleHTML.dat |

Don't Forget to save

Add new response from dropdown

Rule Editor

myRule Test... Save

200_SimpleHTML.dat Match only once



Let's Fiddle stuff!

Exercise 1: Replace a 200 response with a 500 error

Pattern: We want to test how the front end handles an error

Hint: use any page or ntd.codeandtest.org for replacing

Hint: example error responses can be seen at <http://getstatuscode.com/500>



Let's Fiddle stuff!

Exercise 2: Replace an image with another image

Pattern: We want to switch out any file (such as a new version of a css file or javascript library) to see how it works in production

Hint: use any page or the image page at ntd.codeandtest.org



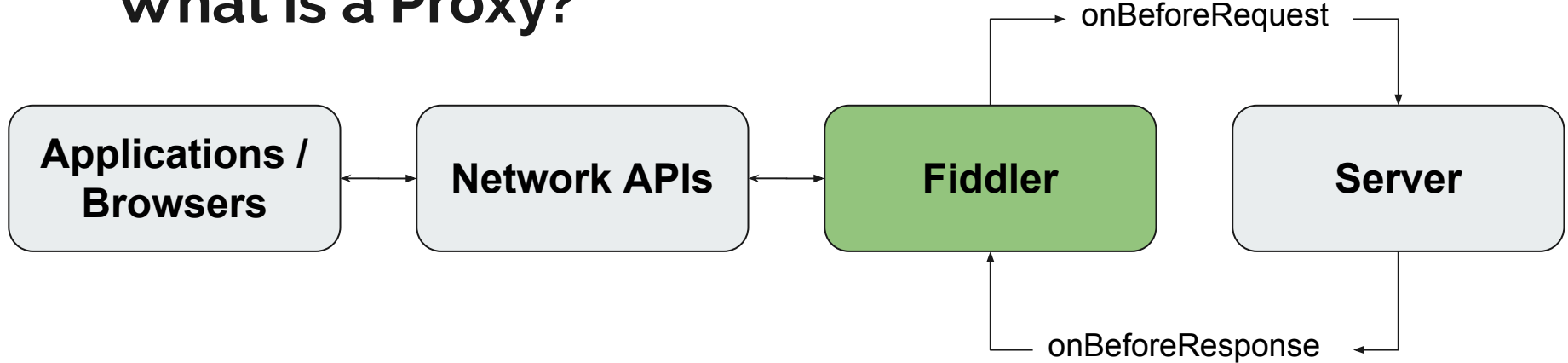
Any questions so far?



Custom Rules / FiddlerScript

- It's pretty much javascript!
 - Except it's completely different
- <https://docs.telerik.com/fiddler/KnowledgeBase/FiddlerScript/>
- We mostly operate inside two existing methods:
 - OnBeforeRequest
 - OnBeforeResponse
- Note: ctrl+s does not work if opened from right hand menu

What is a Proxy?



FiddlerScript

```
if (oSession.host == "codeandtest.org"  
    && oSession.url.Contains("hello"))  
{  
    oSession["ui-color"] = "purple";  
    oSession["ui-bold"] = "test";  
}
```



Let's Fiddle stuff!

Exercise 1: Change the session color for a particular url or host/path

Exercise 2: Rewrite the request url to point at another url

Exercise 3: Rewrite the response code

Exercise 4: Rewrite the response body

Exercise 5: Add a delay to the request or response



Let's Fiddle stuff!

Exercise 6: Variable delay - make all requests with a specific path take between 0 and 10 seconds longer than they should.

Exercise 7: Intermittent failure recovery - make 10% of requests return an error and an empty body.

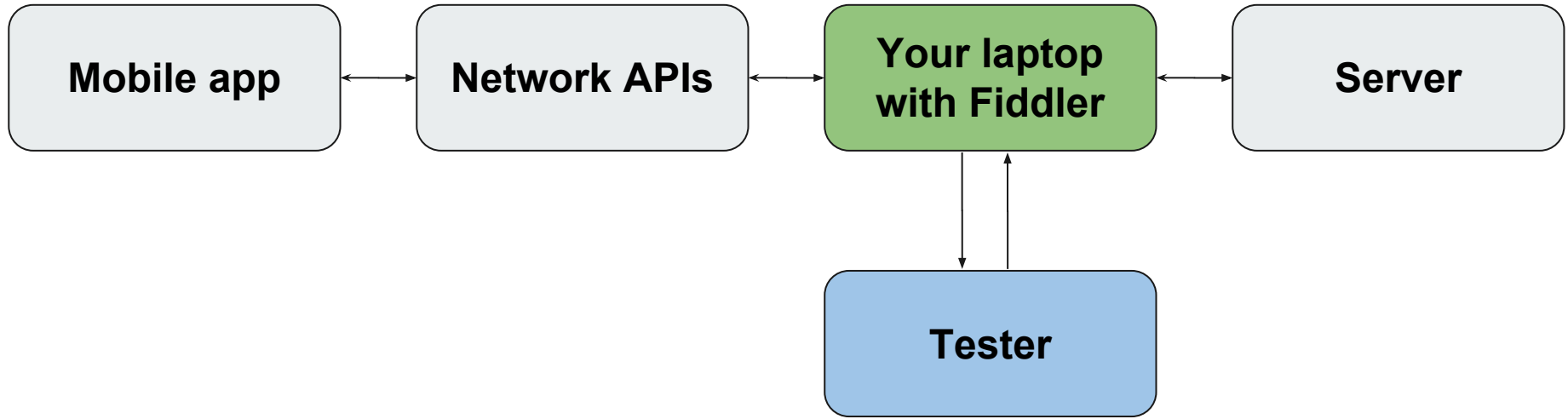


Proxy for other devices




- Setting it up
- Android limitations
- iOS limitations



What is a Proxy?




Proxy for other devices

No SIM  22:01  92% 

[← Nurk](#) **Configure Proxy** [Save](#)

Off

Manual 

Automatic

Server 192.168.1.68

Port 8888

Authentication

Options



General

HTTPS

Connections

Gateway

Appearance

Scripting

Extensions

Performance

Tools

Fiddler can debug traffic from any application that accepts a HTTP Proxy. All WinINET traffic is routed through Fiddler when "File > Capture Traffic" is checked.

[Learn more...](#)Fiddler listens on port: [Copy Browser Proxy Configuration URL](#)

- Capture FTP requests
- Allow remote computers to connect
- Reuse client connections
- Reuse server connections

 Act as system proxy on startup Monitor all connections Use PAC Script DefaultLAN

Bypass Fiddler for URLs that start with:

[Help](#)

Note: Changes may not take effect until Fiddler is restarted.

OK

Cancel



Let's Fiddle stuff?

Exercise: Point your smartphone at Fiddler running on your laptop



Let's discuss your use cases!



Questions



Thank you!