# Debugging

# Outline

- General debugging
- Performance debugging
- Help Us Support You Better

# Scientific method of debugging

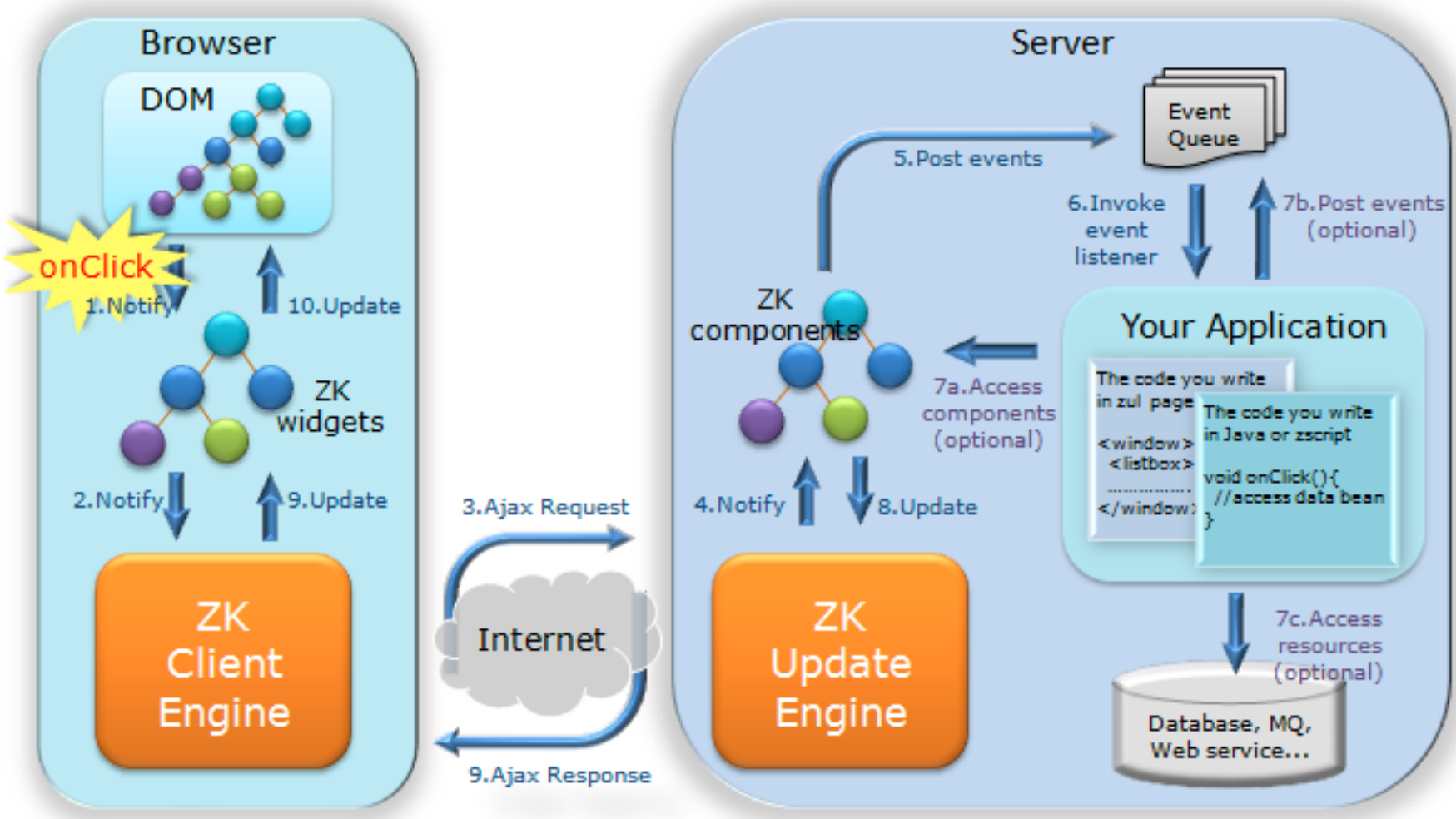
- Stabilize the error
  - Find a simplest case to reproduce the error
- Locate the source of the error
  - Gather the data that produces the defect
  - Analyze the data and form a hypothesis about the defect
  - Prove or disprove the hypothesis
- Fix the defect
- Test the fix
- Look for similar errors

Extracted from “Code Complete”, Steve McConnel

# Stabilize the error

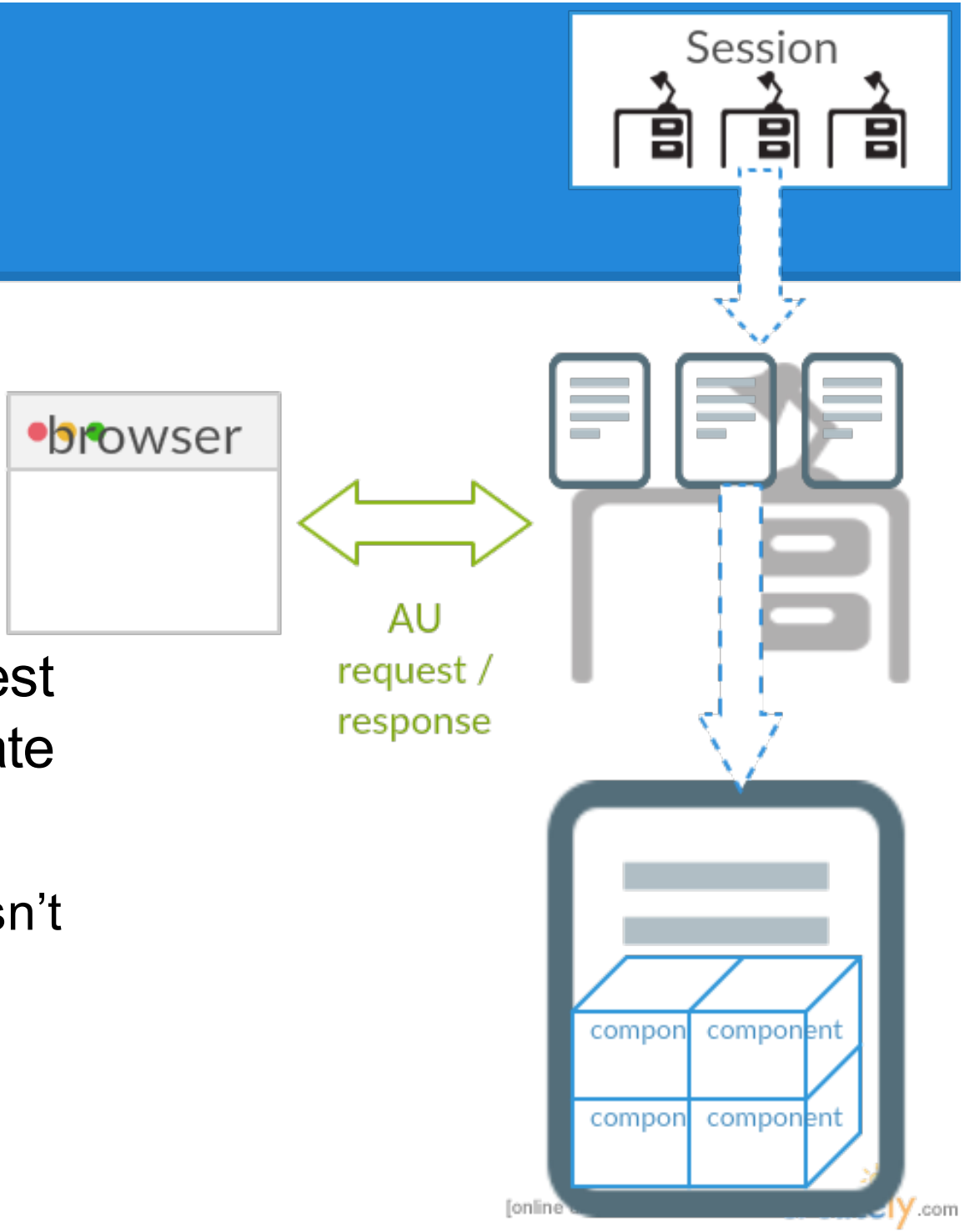
- Find a repeatable way to reproduce it
  - Not-repeatable - Timing issue
- Narrow down the problem
  - to reduce the potential factors that cause the problem
  - More factors, more hypothesis to test
  - Reduce components
  - Reduce steps to reproduce
  - Reduce application logic

# A brief review of ZK architecture



# Scope

- One Session has desktops
- One desktop has pages
  - Each HTTP request to a zul page create a desktop
  - An AU request doesn't create a desktop
- One page has components



# Scope

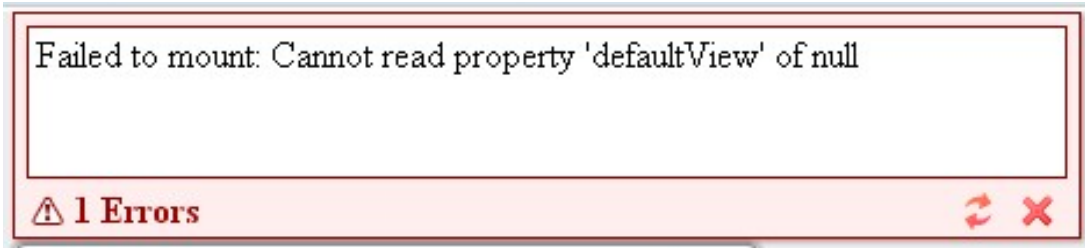
- A basic scope in ZK is “desktop”
  - a logic scope defined by ZK
  - You can treat the scope as a browser tab in the most cases.
- each HTTP request to a zul page creates a corresponding desktop object
  - all subsequent ajax requests are sent to this desktop
  - Ajax request won’t change a browser’s URL
- ZK components, controller (composer, ViewModel) are in desktop scope
  - Reload a URL, recreate UI components

# Locate the source of the error

- A common error message
  - java.lang.IllegalStateException: Access denied: component, <Listcell z_27_b53>, belongs to another desktop: [Desktop g272]
- Root cause:
  - A variable in a composer declared as static
  - A component is passed by event queue to another desktop to access

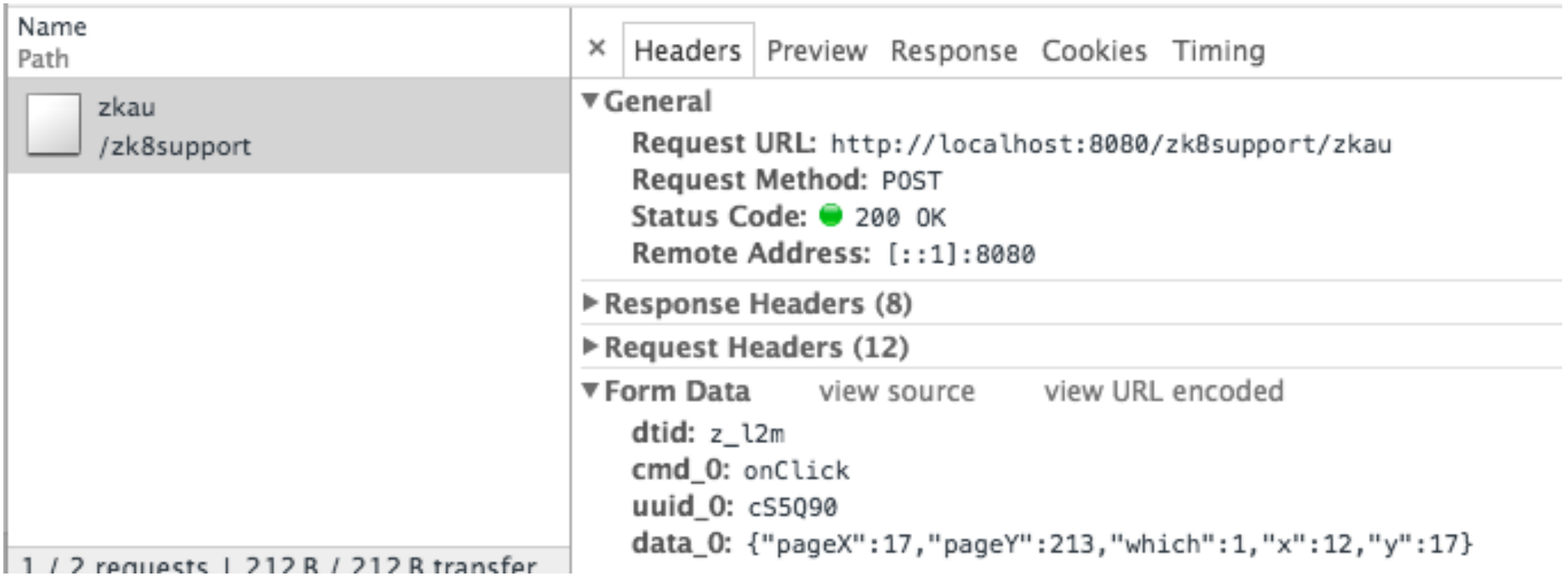
# Locate the source of the error

- Usually it’s a component internal js error



# AU request

- Asynchronous Update (AJAX)



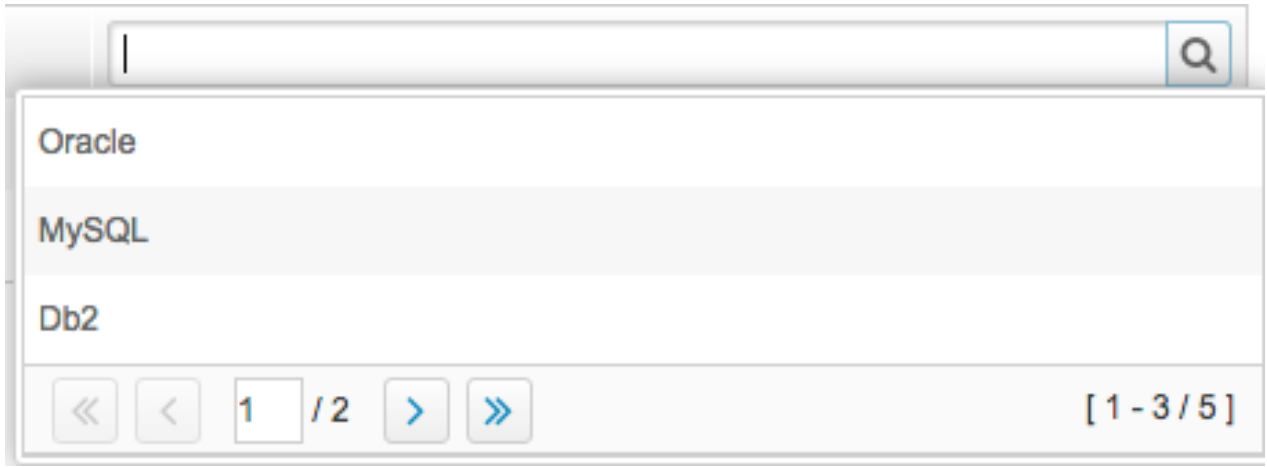

- Handled by org.zkoss.zk.ui.http.DHtmlUpdateServlet

# AU request

```
dtid:z_l2m
cmd_0:onClick
uuid_0:cS5Q90
data_0:{"pageX":17,"pageY":213,"which":1,"x":1
2,"y":17}
```

- dtid, Desktop id
- cmd, event name
- uuid, target component’s uuid
- data, event related data, depending on the event

# AU request

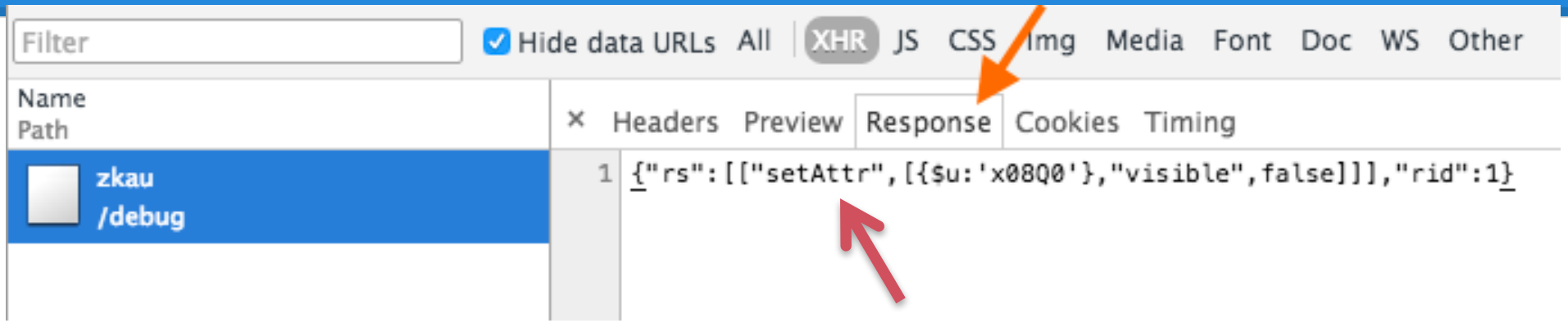



```
dtid:z_jvf
cmd_0:onOpen
uuid_0:zY5Q6
data_0:{“open”:true, “value”:””}
```

# AU request

- When to send
  - a non-deferrable event
    - onRender, onDataLoading
    - sent immediately
  - An important event
    - onChange
    - Sent even no listener registered (with other events)
  - a deferrable event listener registered on the server
    - queued at the client
    - sent with other non-deferrable events
  - If non of the above case
    - No event listener registered
    - dropped

# AU Response


Filter
Hide data URLs
All
XHR
JS
CSS
Img
Media
Font
Doc
WS
Other
Name
Path
zkau
/debug
Headers
Preview
Response
Cookies
Timing
1
{"rs":[["setAttr",[{$u:'x08Q0'},"visible",false]]],"rid":1}

# AU Response

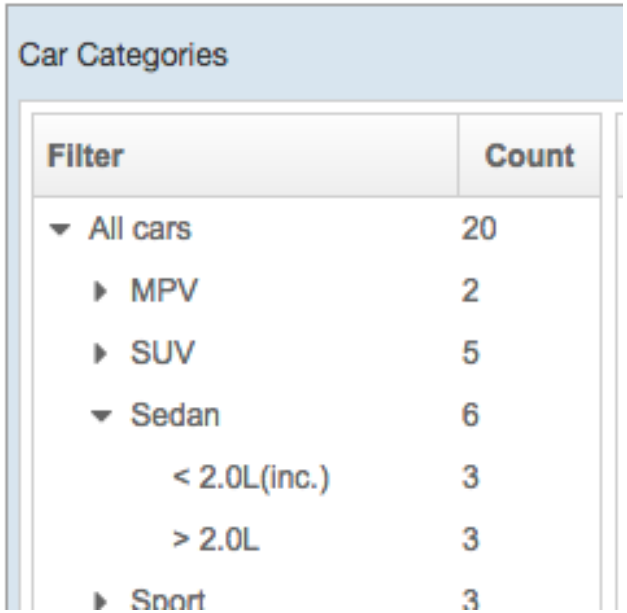
```
{"rs":[["outer",[{$u:'yZ6Q2'},[
['zul.tab.Tabbox','yZ6Q2',{width:'20%',prolog:'\n\t'},{},[
['zul.tab.Tabs','yZ6Q3',{},{},[
['zul.tab.Tab','yZ6Q8',{$onSelect:false,$onClose:true,label
:'tab 1',iconSclass:'z-icon-book',selected:true},{},[]],
['zul.tab.Tab','yZ6Qf',{$onSelect:false,$onClose:true,label
:'tab 2',iconSclass:'z-icon-book'},{},[]]]],
…],"rid":1}
```

# AU Response


```
{"rs":[["rm",["zY5Q10"]],["rm",["zY5Qz"]],["rm",[
"zY5Q60"]],["rm",["zY5Q80"]],["rm",["zY5Qa0"]],["
rm",["zY5Qe0"]],["rm",["zY5Qc0"]],["rm",["zY5Qq0"
]],["rm",["zY5Qo0"]],["rm",["zY5Qm0"]],…}
```

# AU Response


```
{"rs":[["addChd",["iB0Zk0",[
['zul.sel.Treeitem','iB0Zy1',{open:false,_loaded:true},{},[
['zul.sel.Treerow','iB0Zz1',{},{},[
['zul.sel.Treecell','iB0Z_2',{label:'< 2.0L(inc.)'},{},[]],
['zul.sel.Treecell','iB0Z02',{label:'3'},{},[]]]]]],…,"rid
":7}
```

# MVVM debugging message

- Library property - org.zkoss.bind.DebuggerFactory.enable

# Case Study 1

- Not-working Timer
  - When an unexpected thing happens, observe AU /request / response first

# Case Study 2

- The closing Tabbox
  - After adding a new tab, an open tab always closes itself
  - Observe the au response to find “outer”
  - Re-rendering causes the closing.
  - Can replace List with ListModelList
    - A ListModelList can inform a component when its data changes e.g. add() or remove() is called
    - The component only renders the differential parts instead of rendering all

# Case Study 3

- Unable to select a DB system
  - Select an item with a keyboard
  - Check the listener to select an item
  - We don’t suggest mixing MVC and MVVM approaches unless you are familiar with both approaches.

# Performance Debugging

- Mostly bottleneck can one of the following:
  - Client
  - Network
  - Server

# Performance Debugging

- Network Timing

**Initial Connection / Connecting**

Time it took to establish a connection, including TCP handshakes/retries and negotiating a SSL.

**SSL**

Time spent completing a SSL handshake.

**Request Sent / Sending**

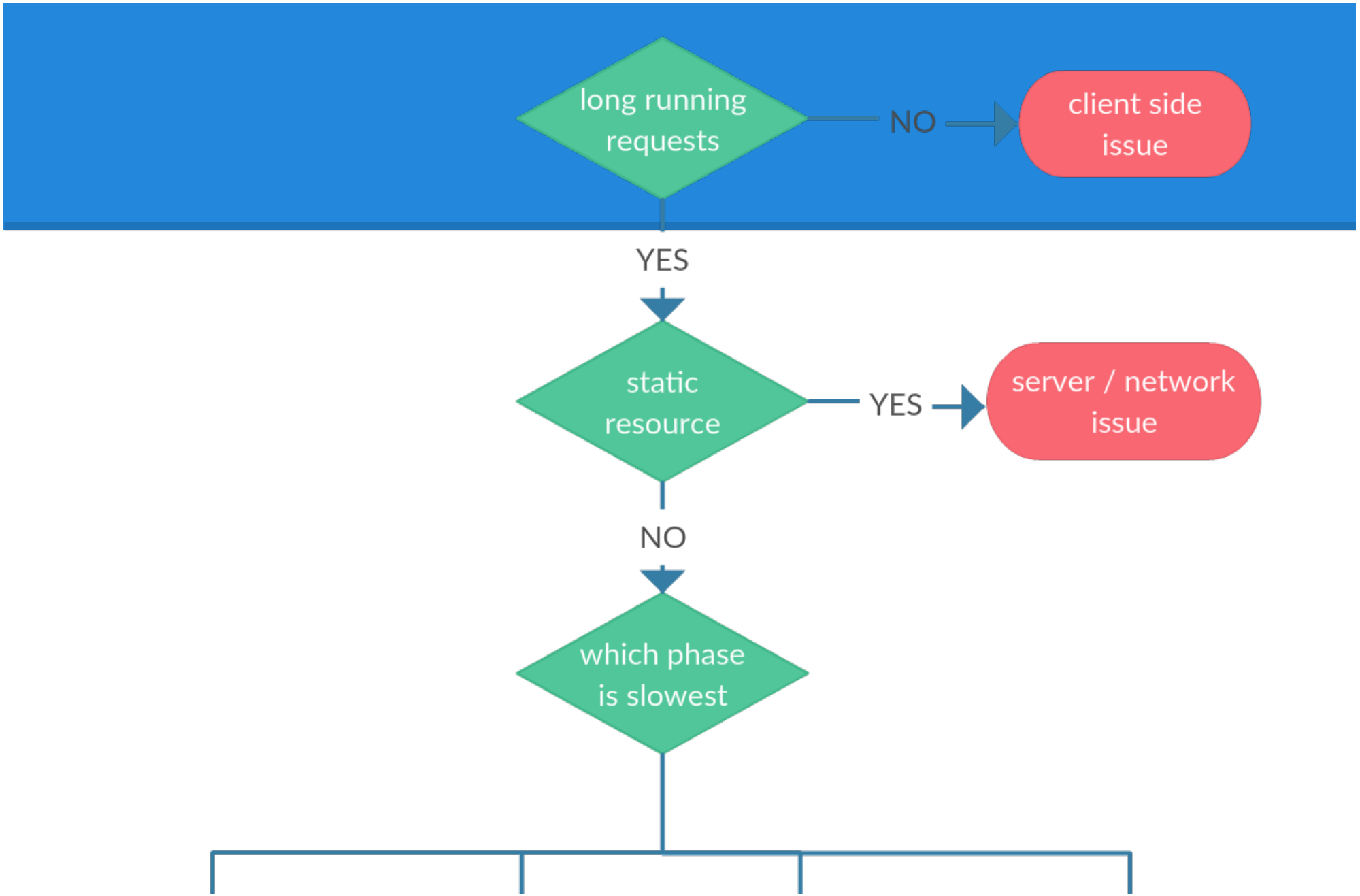
Time spent issuing the network request. Typically a fraction of a millisecond.

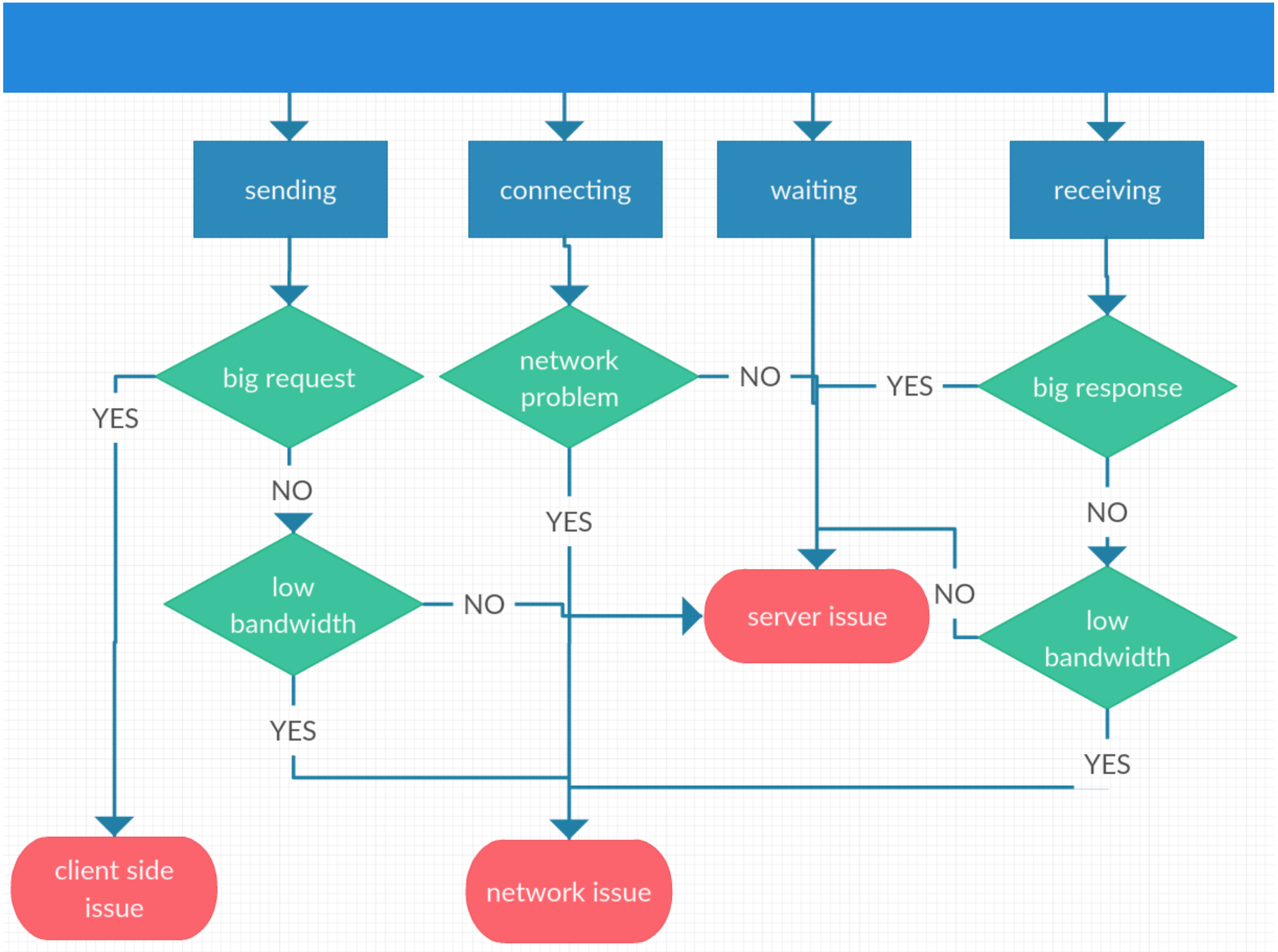
**Waiting (TTFB)**

Time spent waiting for the initial response, also known as the Time To First Byte. This time captures the latency of a round trip to the server in addition to the time spent waiting for the server to deliver the response.

**Content Download / Downloading**

Time spent receiving the response data.


long running requests
NO
client side issue
YES
static resource
YES
server / network issue
NO
which phase is slowest


sending
connecting
waiting
receiving
big request
network problem
big response
low bandwidth
low bandwidth
server issue
client side issue
network issue
YES
NO
YES
NO
YES
NO
NO
YES
NO
YES

# Case Study

- form1.zul
- form2.zul

# Exercise

- performance / simple.zul
- performance / performance.zul
- performance / tree.zul

# Reference

- For complete explanation, please refer tohttp://books.zkoss.org/wiki/ZK_Developer%27s_Reference/Performance_Monitoring/Step_by_Step_Trouble_Shooting

# Help Us Support You Better

- A runnable example is the best for us. You can:
  - Use zkfiddle
  - Provide a zul and its related Java source
  - Zip your maven project with pom.xml and include necessary source code
  - Provide a public URL that we can access
  - Remember to turn on <debug-js>true</debug-js> in your zk.xml

# Help Us Support You Better

- Tips for creating a runnable sample:
  - Create a test ZUL file as a sample and move your Java codes (mostly event listeners) into <zscript>
  - Replace your DB data with static sample data in order to run without a database
  - For UI which is built by API
    - a javascript to convert from widgets to a zul

# Help Us Support You Better

- If you cannot provide a reproducible sample
  - <debug-js>true</debug-js>
  - Send us the error message
  - Au request
  - Au response