

10 Design Tips for MicroServices Developers

Development and Design sits at the intersection of
Science, Art, Methods, and Patterns

Jim Tyrrell

Senior Principal Solutions Architect

Member of the Office of Chief Technology NAPS

The Designatic

Founder of Design 4 Developers (<https://design4developers.io>)

10 Design Tips for MicroServices Developers

Government Centered Resources Design Thinking

<https://itk.mitre.org/>

MITRE'S INNOVATION TOOLKIT

MITRE's Innovation Toolkit is a collection of proven and repeatable problem-solving methods to help you and your team do something different that makes a difference.

GET OUR BOOK

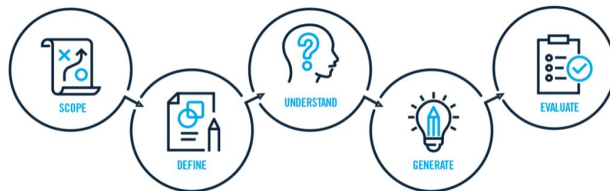
The **Toolbox of Innovation** puts the reader in the driver's seat and takes you on an imaginary journey of exploration.

[SEE BOOK DETAILS](#)

TOOLS

The Innovation Toolkit (ITK) is a publicly available, handpicked collection of proven approaches and methods to help your team be more innovative. Derived from human-centered design practices, the toolkit helps teams think creatively, frame problems, build consensus, and ask the right questions. Use ITK to spark insightful conversations, capture meaningful results, and guide creative problem solving.

Click on a circle below to begin exploring the five tool categories, or click the BROWSE button to see the whole list.



[BROWSE ALL TOOLS](#)

<https://openpracticelibrary.com/>

OPEN PRACTICE LIBRARY

The Mobius Loop

An iterative process model for rapidly developing digital products. The Open Practice Library is organized around this model. The buttons below will show you practices that fall on the relevant part of the loop.

*Mobius Loop™ is licensed under CC BY-SA 4.0.

∞ ALL FOUNDATION DISCOVERY OPTIONS DELIVERY

Know the name of the practice?

<h4>10-for-10</h4> <p>The ultimate brainstorming exercise!</p>	<h4>15/5 Reports</h4> <p>Reporting for people who don't have time for reports</p>	<table border="1"><tr><td>Culture design, management, development, innovation, solution practices</td><td>Tools & Technology Platforms, networks, interfaces, infrastructure, applications</td><td>Process Operations, management, support release practices, frameworks</td></tr><tr><td>Experimentation Stakeholder, staff & customer engagement, open plans</td><td>Communication Direction, decision approval cadence, online, meeting cadence, online, offline, 1-to-1 to many, fast feedback loops, realtime/async</td><td>Governance Direction, decision approval, checklists, budgeting, report values and metrics</td></tr></table>	Culture design, management, development, innovation, solution practices	Tools & Technology Platforms, networks, interfaces, infrastructure, applications	Process Operations, management, support release practices, frameworks	Experimentation Stakeholder, staff & customer engagement, open plans	Communication Direction, decision approval cadence, online, meeting cadence, online, offline, 1-to-1 to many, fast feedback loops, realtime/async	Governance Direction, decision approval, checklists, budgeting, report values and metrics
Culture design, management, development, innovation, solution practices	Tools & Technology Platforms, networks, interfaces, infrastructure, applications	Process Operations, management, support release practices, frameworks						
Experimentation Stakeholder, staff & customer engagement, open plans	Communication Direction, decision approval cadence, online, meeting cadence, online, offline, 1-to-1 to many, fast feedback loops, realtime/async	Governance Direction, decision approval, checklists, budgeting, report values and metrics						

10 Design Tips for MicroServices Developers

Who am I?

Developer and Designer

Development = Science, Art, Patterns, and Methods

10 Design Tips for MicroServices Developers

Who am I?

Developer and Designer

Development = Science, Art, Patterns, and Methods

Design = Art, Patterns, Methods, and Science

10 Design Tips for MicroServices Developers

Who am I?

Developer and Designer

Development = Science, Art, Patterns, and Methods

Design = Art, Patterns, Methods, and Science

Design is Development

And

Development is Design

10 Design Tips for MicroServices Developers

Who am I?

Developer and Designer

Development = Science, Art, Patterns, and Methods

Design = Art, Patterns, Methods, and Science

Design is Development

And

Development is Design

They are like two objects intertwined by Gravity!!

10 Design Tips for MicroServices Developers

Who is software for?

Humans

“Unless you are impossibly lucky, all software at some point will be consumed by humans, for sure during its development, but also for the years after it was delivered, as it will be maintained and may continue to be used by humans.”

The Designatic

10 Design Tips for MicroServices Developers

What is a User Interface or UI?

At least 3 in the context of software

1. GUI as a GUI is a UI
2. API as Code is a UI
3. API as a Service is a UI

10 Design Tips for MicroServices Developers

The 5Es

Every ~~Great~~ Experience has 3 Organic ~~Curated~~ States

-
- Enter
- Engage
- Exit
-

10 Design Tips for MicroServices Developers

The 5Es

Every Great Experience has 5 ~~Organic~~ Curated States

- Entice

10 Design Tips for MicroServices Developers

The 5Es

Every Great Experience has 5 ~~Organic~~ Curated States

- Entice
- Enter

10 Design Tips for MicroServices Developers

The 5Es

Every Great Experience has 5 ~~Organic~~ Curated States

- Entice
- Enter
- Engage

10 Design Tips for MicroServices Developers

The 5Es

Every Great Experience has 5 ~~Organic~~ Curated States

- Entice
- Enter
- Engage
- Exit

10 Design Tips for MicroServices Developers

The 5Es

Every Great Experience has 5 ~~Organic~~ Curated States

- Entice
- Enter
- Engage
- Exit
- Extend

10 Design Tips for MicroServices Developers

The 5Es

Every Great Experience has 5 ~~Organic~~ Curated States

- Entice
- Enter
- Engage
- Exit
- Extend

All Journeys Should Have Consistency

10 Design Tips for MicroServices Developers

10 Usability Heuristics for User Interface Design

Nielsen Norman Group

1. Visibility of system status
2. Match between system and the real world
3. User control and freedom
4. Consistency and standards
5. Error prevention
6. Recognition rather than recall
7. Flexibility and efficiency of use
8. Aesthetic and minimalist design
9. Help users recognize, diagnose, and recover from errors
10. Help and documentation

10 Design Tips for MicroServices Developers

Making Things that Matter

9 Purpose Defining Statements for Software Development

1. Security over Slyness
2. Scale over Shatter
3. Standardization over Straying
4. Support over Stumbling
5. Sense over Summon
6. Safety over Setback
7. Stewardship over Suffering
8. Statistics & Study over Speculation
9. Status over Surmising

10 Design Tips for MicroServices Developers

10 Design Tips for MicroServices Developers

10 Design Tips for MicroServices Developers

#1 Doc Debris - Entice

- Anti-QuickStarts
- Wall of Intertwined Text
- Missing Steps
- Back and Forth

10 Design Tips for MicroServices Developers

#1 Doc Debris - Entice

Anti-QuickStarts

Pages of Text

w/10s or even 100s of Steps

Getting Started Experiences Labeled as Quickstarts that are anything but quick or evening starting.

10 Design Tips for MicroServices Developers

#1 Doc Debris - Entice

Wall of Intertwined Text

Starts with Step 1

- Windows Instructions
- Linux Instructions
- Mac Instructions

Then Step 2

- Linux Instructions
- Windows Instructions

Then Step 3

- Mac Instructions

Except they are never labeled Step 1, 2, and 3

10 Design Tips for MicroServices Developers

#1 Doc Debris - Entice

Missing Steps

1. Download this Software Package
2. Install w/ these Commands
3. Run It

Then see an error

Are your download and install instructions complete?

10 Design Tips for MicroServices Developers

#1 Doc Debris - Entice

Back and Forth

Open Up This Page of that Guide

Open Up That Page of this Guide

Hope to find a Stackoverflow Article

Use your cheatsheet of cheat codes and instructions

10 Design Tips for MicroServices Developers

#2 Security over Slyness - Entice

- https & tls
- Tokens & Keys
- Password Management
- Defined Timeouts
- Saving Data/FIPS 140-2/3
- Resetting Passwords

10 Design Tips for MicroServices Developers

#2 Security over Slyness - Entice

Tokens/Keys

Where and how do your users discover these keys and how do they expire?

If they expire do you tell your users that?

10 Design Tips for MicroServices Developers

#3 Scale over Shatter - Enter

- Disaster Recovery
- High Availability
- Scalable Infrastructure
 - Knative, Serverless, FaaS
- Saga Pattern & Idempotency

What is your plan to bend but not break?

10 Design Tips for MicroServices Developers

#4 Support over Stumbling - Enter

- Who are you targeting?
- Can they be successful?

Self Service and Self Starting

10 Design Tips for MicroServices Developers

#4 Support over Stumbling - Enter

```
Message message = Message.creator(  
    new com.twilio.type.PhoneNumber("+15558675310"),  
    new com.twilio.type.PhoneNumber("+15017122661"),  
    "This is the ship that made the Kessel Run in four  
    .create();
```

10 Design Tips for MicroServices Developers

#4 Support over Stumbling - Enter

```
Message message = Message.creator(  
    new com.twilio.type.PhoneNumber("+15558675310"),  
    new com.twilio.type.PhoneNumber("+15017122661"),
```

```
Exception in thread "main" com.twilio.exception.ApiException: The From phone number +1  
7208392251 is not a valid, SMS-capable inbound phone number or short code for your acc  
ount.
```

```
    at com.twilio.rest.api.v2010.account.MessageCreator.create(MessageCreator.java  
:530)
```

```
    at com.twilio.rest.api.v2010.account.MessageCreator.create(MessageCreator.java  
:25)
```

```
    at com.twilio.base.Creator.create(Creator.java:40)
```

```
    at org.redhat.twillo.SendSMS.sendMessageTest(SendSMS.java:25)
```

```
    at org.redhat.twillo.SendSMS.main(SendSMS.java:50)
```

10 Design Tips for MicroServices Developers

#4 Support over Stumbling - Enter

```
Message message = Message.creator(  
    new com.twilio.type.PhoneNumber("+15558675310"),  
    new com.twilio.type.PhoneNumber("+15017122661"),  
    "This is a test message from Twilio")
```


```
Exception in thread "main" com.twilio.exception.ApiException: The From phone number +1  
7208392251 is not a valid, SMS-capable inbound phone number or short code for your acc  
ount.
```

```
    at com.twilio.rest.api.v2010.account.MessageCreator.create(MessageCreator.java  
:530)
```

```
    at com.twilio.rest.api.v2010.account.MessageCreator.create(MessageCreator.java  
:25)
```

```
    at com.twilio.base.Creator.create(Creator.java:40)
```

```
    at org.redhat.twilio.SendSMS.sendMessageTest(SendSMS.java:25)
```

 **creator**(PhoneNumber to, PhoneNumber from, String

10 Design Tips for MicroServices Developers

#5 Standardization over Straying - Engage

POST /pet Add a new pet to the store

PUT /pet Update an existing pet

GET /pet/{petId} Find pet by ID

POST /pet/{petId} Updates a pet

DELETE /pet/{petId} Deletes a pet

POST /pet/{petId}/uploadImage

10 Design Tips for MicroServices Developers

#5 Standardization over Straying - Engage

POST /pet Add a new pet to the store

PUT /pet/{petId}

GET /pet/{petId} Find pet by ID

POST /pet/{petId} Updates a pet

DELETE /pet/{petId} Deletes a pet

POST /pet/{petId}/uploadImage

Engage

10 Design Tips for MicroServices Developers

#5 Standardization over Straying - Engage

POST /pet Add a new pet to the store

Should it be
ID or
Id?

PUT /pet/{petId}

GET /pet/{petId} Find pet by ID

POST /pet/{petId} Updates a pet

DELETE /pet/{petId} Deletes a pet

POST /pet/{petId}/uploadImage

Engage

10 Design Tips for MicroServices Developers

#5 Standardization over Straying - Engage

```
<id>0</id>
```

Should it be
ID or
Id?

```
/pet/{petId} Find pet by ID
```

```
petId - ID of pet to return
```

10 Design Tips for MicroServices Developers

#5 Standardization over Straying - Engage

Which id is
it?

```
{
  "id": 0,
  "category": {
    "id": 0,
    "name": "
  },
  "tags": [
    {
      "id": 0,
      "name": "string"
    }
  ]
}
```

Engage

10 Design Tips for MicroServices Developers

#6 Sense over Summon - Engage

- Do they see the next step, or do you expect them to summon steps from the depths of knowledge?

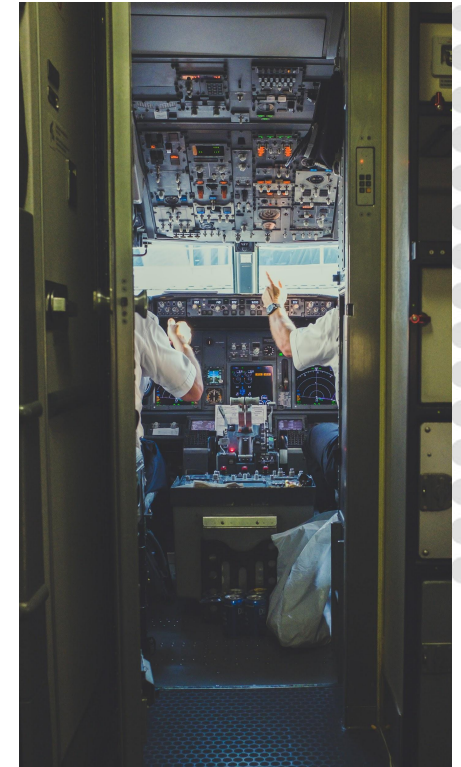
10 Design Tips for MicroServices Developers

#6 Sense over Summon - Engage



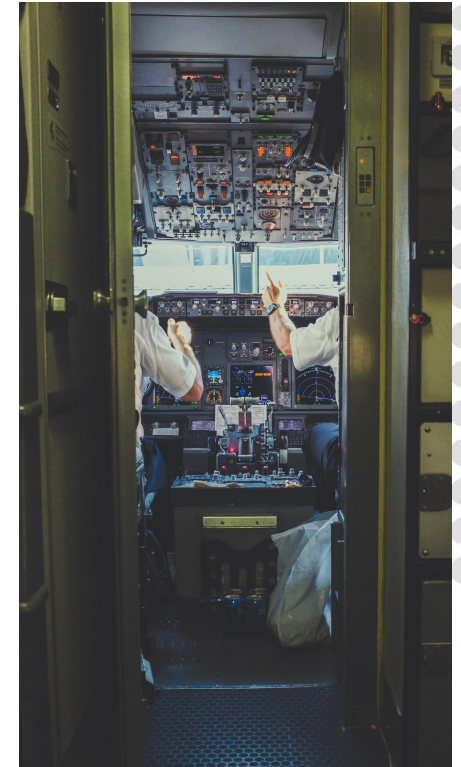
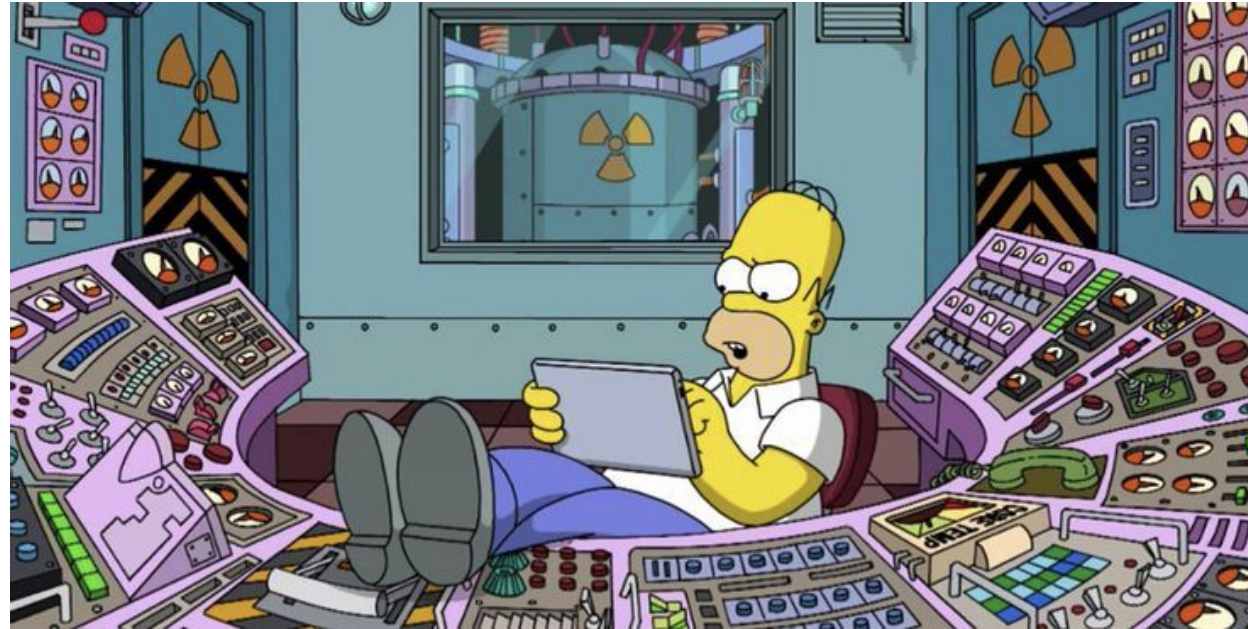
10 Design Tips for MicroServices Developers

#6 Sense over Summon - Engage



10 Design Tips for MicroServices Developers

#6 Sense over Summon - Engage



10 Design Tips for MicroServices Developers

#7 Safety over Setback - Engage

- Back button/Undo
- Eventual Consistency
- Thread Safety
- Idempotency

10 Design Tips for MicroServices Developers

#7 Safety over Setback - Engage

- Error messages give corrective steps and not just that something went wrong.
- Passed in values are validated and incorrect ones are clearly identified with potential fixes.
- A white screen of death is never okay.

10 Design Tips for MicroServices Developers

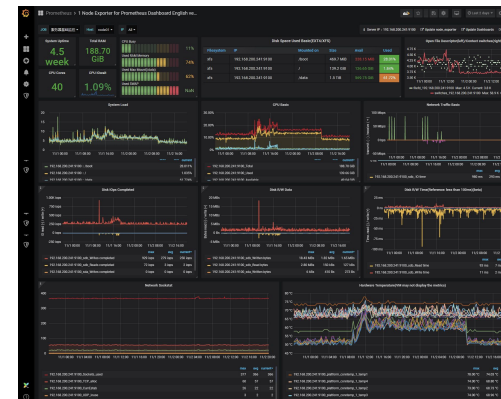
#8 Stewardship over Suffering - Engage

- Continuing and Ongoing Use
- HelloWorld vs Neo-Maxi Super Duper
- Observe your users and remove friction, toil, and invisible work!!

10 Design Tips for MicroServices Developers

#9 Status over Surmising - Exit

- Clear Ending
- Logging
- Availability
- Metrics



10 Design Tips for MicroServices Developers

#10 Statistics & Study over Speculation - Extend

- Throughput or Success Report
- Followup
- Get Users Satisfaction
 - User Interviews and Surveys

10 Design Tips for MicroServices Developers

#10 Statistics & Study over Speculation - Extend

A Story about a Nameless Vendor from a Friend

10 Design Tips for MicroServices Developers

Microservices and UX in Government

Human Centered Design

- 21st Century Integrated Digital Experience Act aka IDEA
 - Less than 2% of Surveyed Forms CompliantBy requiring Faxing or Printing how many citizens are you leaving out?
- Replatforming and Updating Systems
 - Integrated Eligibility Systems Nationwide
- These 10 Tips can help any agency looking to create public or private APIs

10 Design Tips for MicroServices Developers

Microservices and UX in Government

Kessel Run



Airman 1st Class Maxwell Lehmann (right), 2nd Contracting Squadron specialist, works with a group for the Kessel Run project. (Courtesy Photo)

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris
2. Enter - Security over Slyness

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris
2. Enter - Security over Slyness
3. Enter - Scale over Shatter

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris
2. Enter - Security over Slyness
3. Enter - Scale over Shatter
4. Enter - Support over Stumbling

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris
2. Enter - Security over Slyness
3. Enter - Scale over Shatter
4. Enter - Support over Stumbling
5. Engage - Standardization over Straying

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris
2. Enter - Security over Slyness
3. Enter - Scale over Shatter
4. Enter - Support over Stumbling
5. Engage - Standardization over Straying
6. Engage - Sense over Summon

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris
2. Enter - Security over Slyness
3. Enter - Scale over Shatter
4. Enter - Support over Stumbling
5. Engage - Standardization over Straying
6. Engage - Sense over Summon
7. Engage - Safety over Setback

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris
2. Enter - Security over Slyness
3. Enter - Scale over Shatter
4. Enter - Support over Stumbling
5. Engage - Standardization over Straying
6. Engage - Sense over Summon
7. Engage - Safety over Setback
8. Engage - Stewardship over Suffering

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris
2. Enter - Security over Slyness
3. Enter - Scale over Shatter
4. Enter - Support over Stumbling
5. Engage - Standardization over Straying
6. Engage - Sense over Summon
7. Engage - Safety over Setback
8. Engage - Stewardship over Suffering
9. Exit - Status over Surmising

10 Design Tips for MicroServices Developers

Summary:

1. Entice - Doc Debris
2. Enter - Security over Slyness
3. Enter - Scale over Shatter
4. Enter - Support over Stumbling
5. Engage - Standardization over Straying
6. Engage - Sense over Summon
7. Engage - Safety over Setback
8. Engage - Stewardship over Suffering
9. Exit - Status over Surmising
10. Extend - Statistics & Study over Speculation

10 Design Tips for MicroServices Developers

In Conclusion:

Gives you a Framework

10 Design Tips for MicroServices Developers

In Conclusion:

Gives you a Framework
To Remove
Friction, Toil, and Invisible Work

10 Design Tips for MicroServices Developers

In Conclusion:

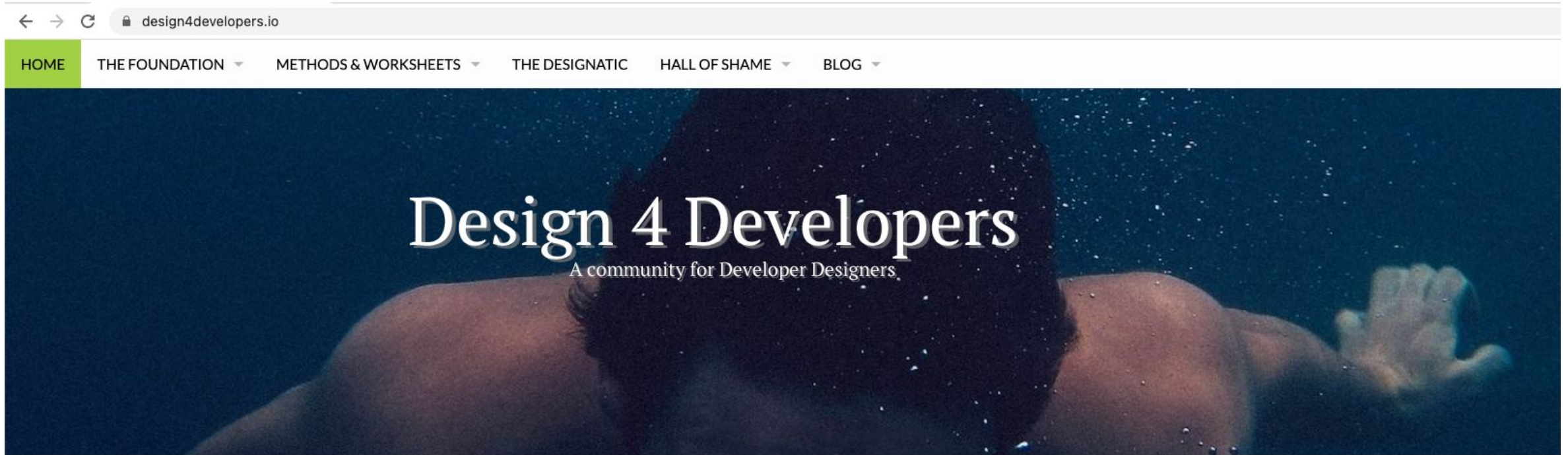
Gives you a Framework
To Remove
Friction, Toil, and Invisible Work

When you do this you will make your users smarter!!!

10 Design Tips for MicroServices Developers

Design 4 Developers

A home for Developers, Designers, Doers, and Dreamers



<https://des4.dev>

Extend

10 Design Tips for MicroServices Developers

Join Us:

Design4Developers.io

FaceBook

Twitter

LinkedIn

Instagram

Let's Chat

Monthly Workshop

4th Tuesday of the Month (For awhile)

Next one is Tuesday, September 28th, 2021

des4.dev

des4.dev/fb

des4.dev/tw

des4.dev/li

des4.dev/ig

des4.dev/meet

des4.dev/workshop

Extend - Thank You!!