More Testable Code

With Hexagonal Architecture

Visit r2ha.com

Ted M. Young
Java Trainer, Coach
& Live Coder

✓ TESTABLE ✗

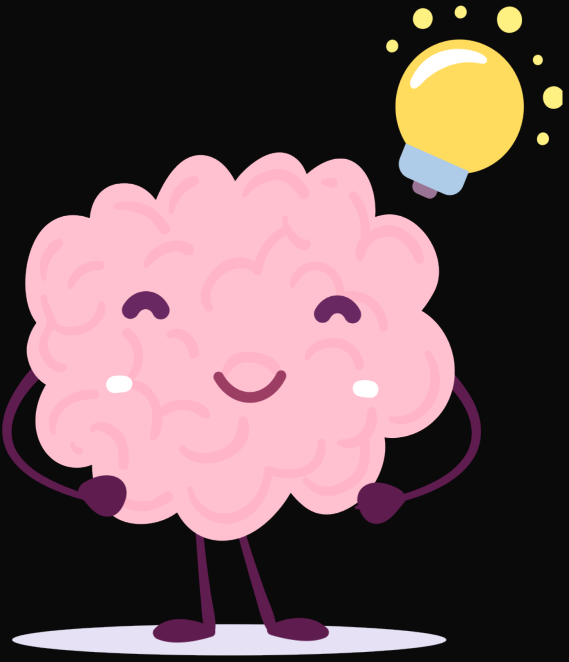
HEXAGONAL ARCHITECTURE

ORGANIZED

Twitter: @JitterTed
Email: ted@tedmyoung.com
Live Coding: https://JitterTed.Stream

Web: https://TED.dev
YouTube: https://JitterTed.TV

Ask Questions As You Need



What are

Tests?

Tests Are

- Isolated
- Automatic
- Concise
- Self-Verifying
- Precise
- Predictable
- Written by developers

Unit Test? Integration Test?

Nope.

Why is writing tests so hard? 😣

what do you think?

Writing tests is hard because…

Classes poorly organized	Classes directly access I/O
Too much state	Infrastructure
Too many responsibilities	Hardware
	Framework

SCION-T

Separating
Concerns of
I/O and
Non-I/O for
Testability

I/O is…

Anything outside of the current process

I/O includes…

- Hardware
  - System clock, Random number generator, Files
- Remote Services
  - via HTTP, Queues, RPC
- Databases
  - Including "in-memory" databases (H2, SQLite, Redis)
- Other Processes
- Frameworks

Why do tests care about dependencies on I/O?

SPEED, EASE, & PREDICTABILITY

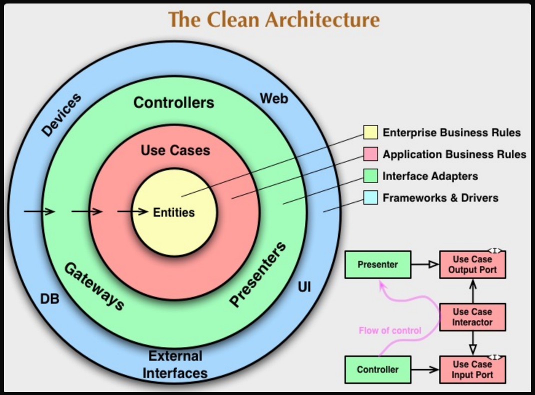
Two Kinds of Tests

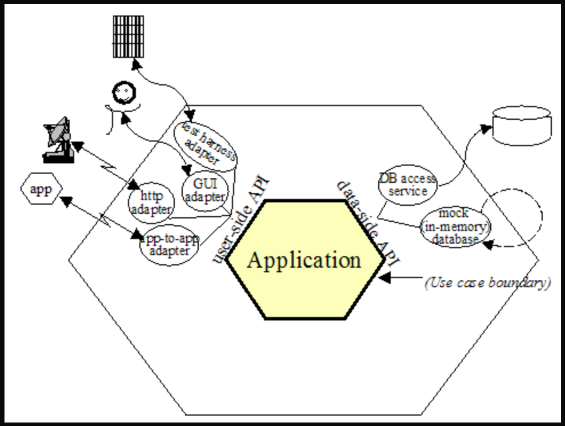
I/O

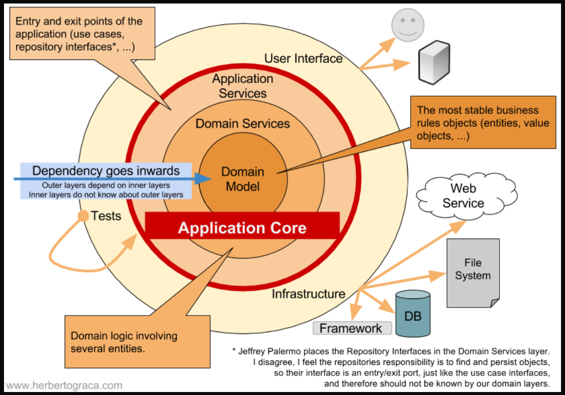
I/O-Free*

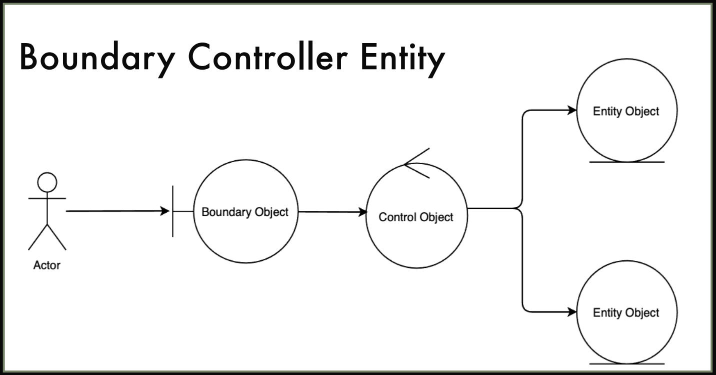
* These are practically "free" to execute

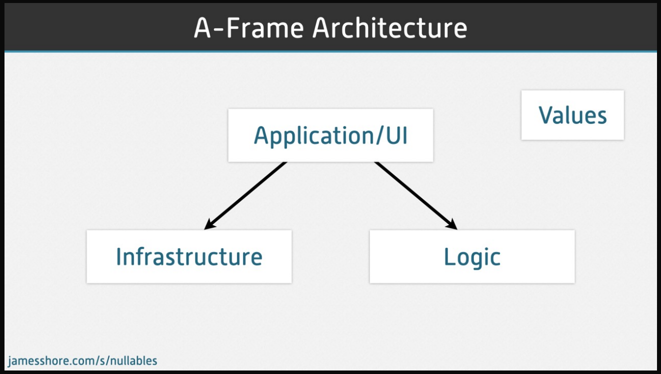
SCION-T Architectures





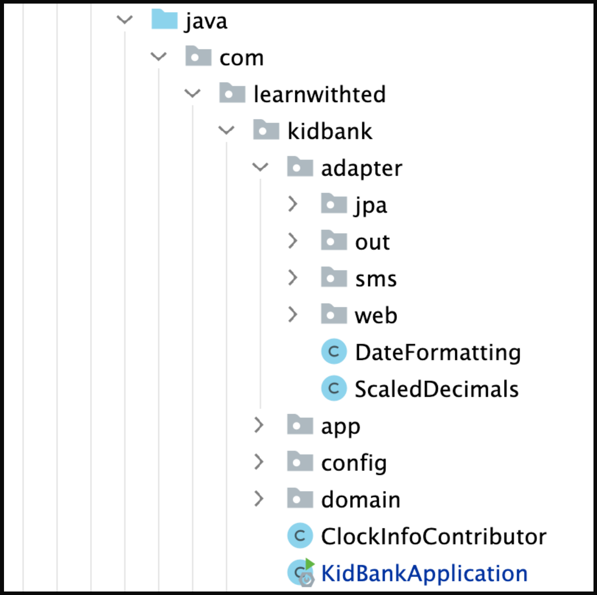

Entry and exit points of the application (use cases, repository interfaces*, ...)
User Interface
Application Services
Domain Services
Domain Model
Dependency goes inwards
Outer layers depend on inner layers
Inner layers do not know about outer layers
Tests
Application Core
The most stable business rules objects (entities, value objects, ...)
Web Service
File System
DB
Infrastructure
Framework
Domain logic involving several entities.
* Jeffrey Palermo places the Repository Interfaces in the Domain Services layer.
I disagree, I feel the repositories responsibility is to find and persist objects,
so their interface is an entry/exit port, just like the use case interfaces,
and therefore should not be known by our domain layers.
www.herbertograca.com





Architecture is…

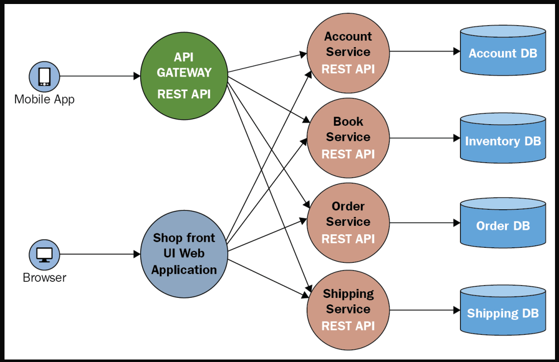
Organization of Code

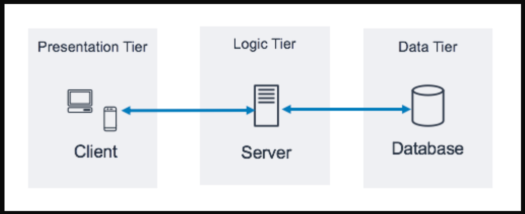


```
java
  com
    learnwithted
      kidbank
        adapter
          jpa
          out
          sms
          web
          DateFormatting
          ScaledDecimals
        app
        config
        domain
        ClockInfoContributor
        KidBankApplication
```

layers

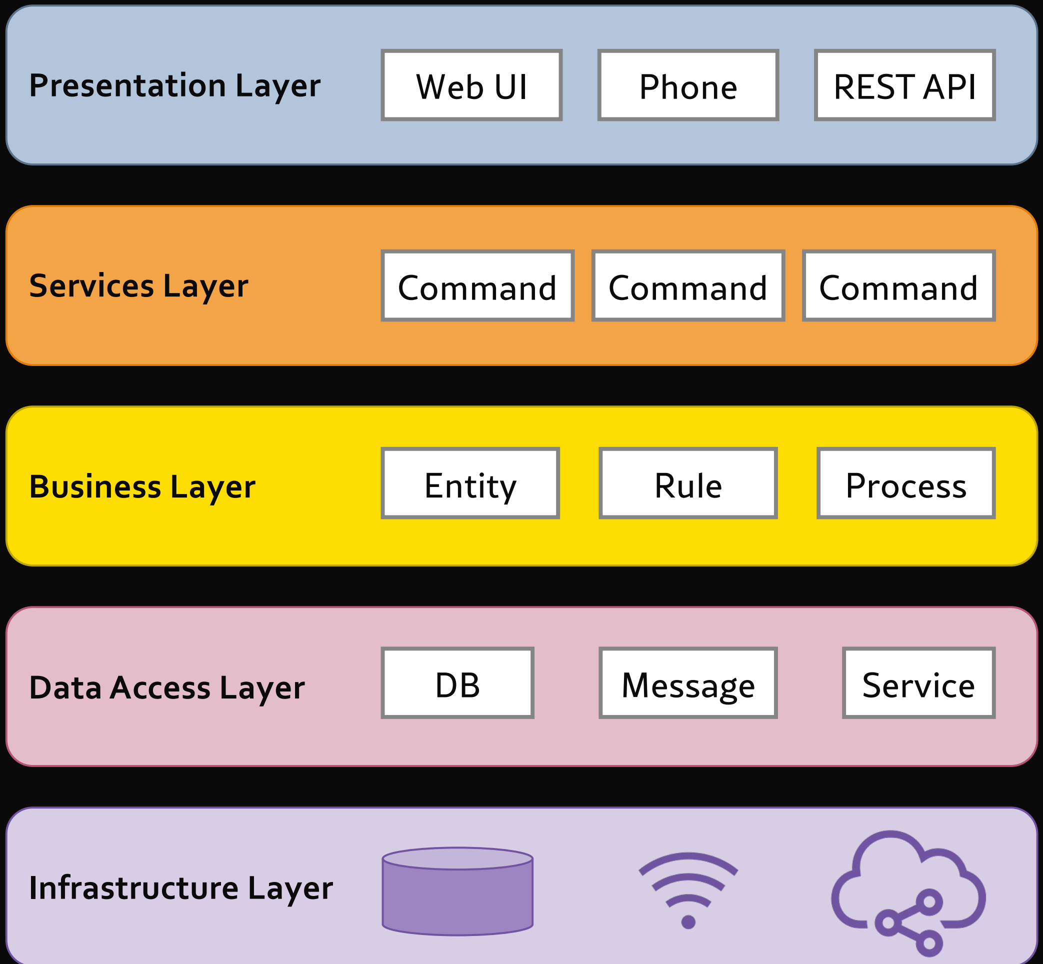
Deployment/Operational



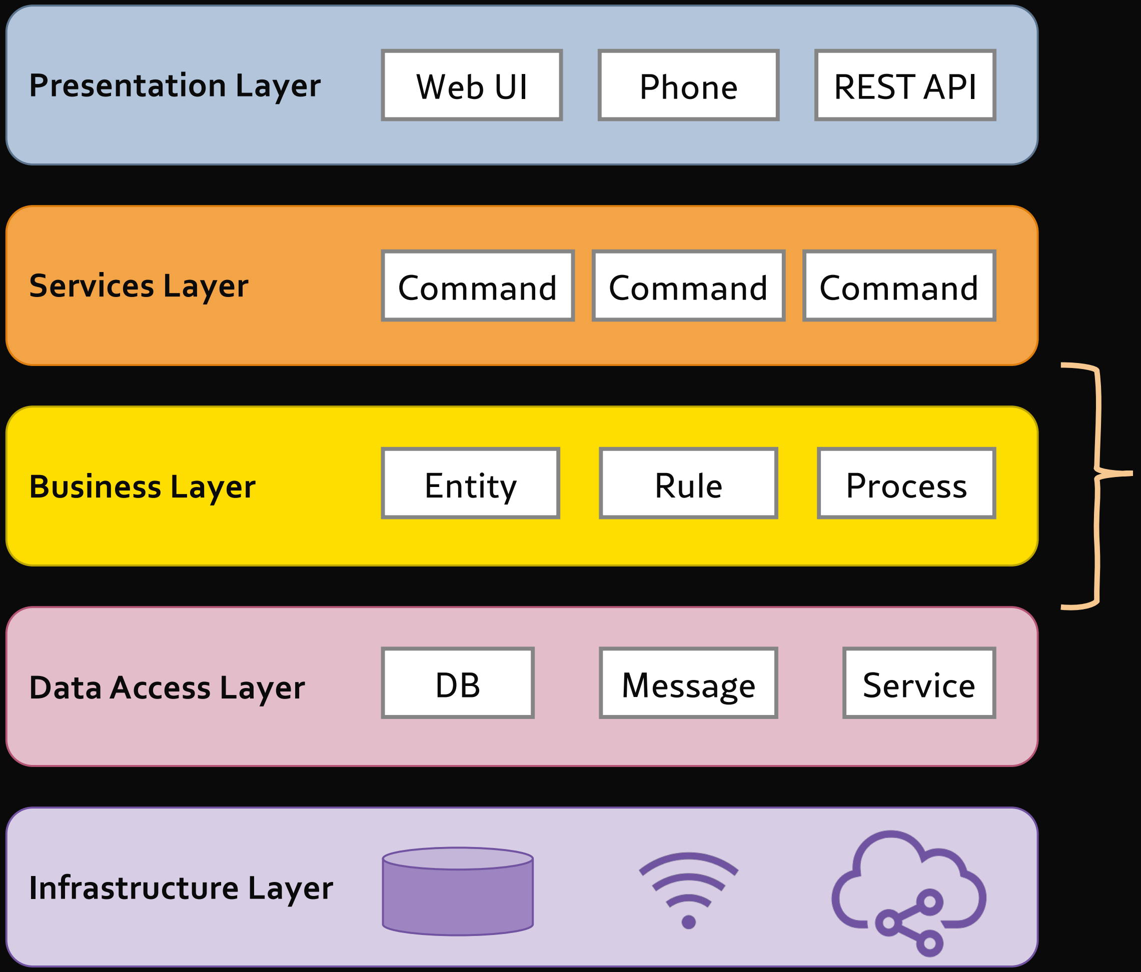


tiers

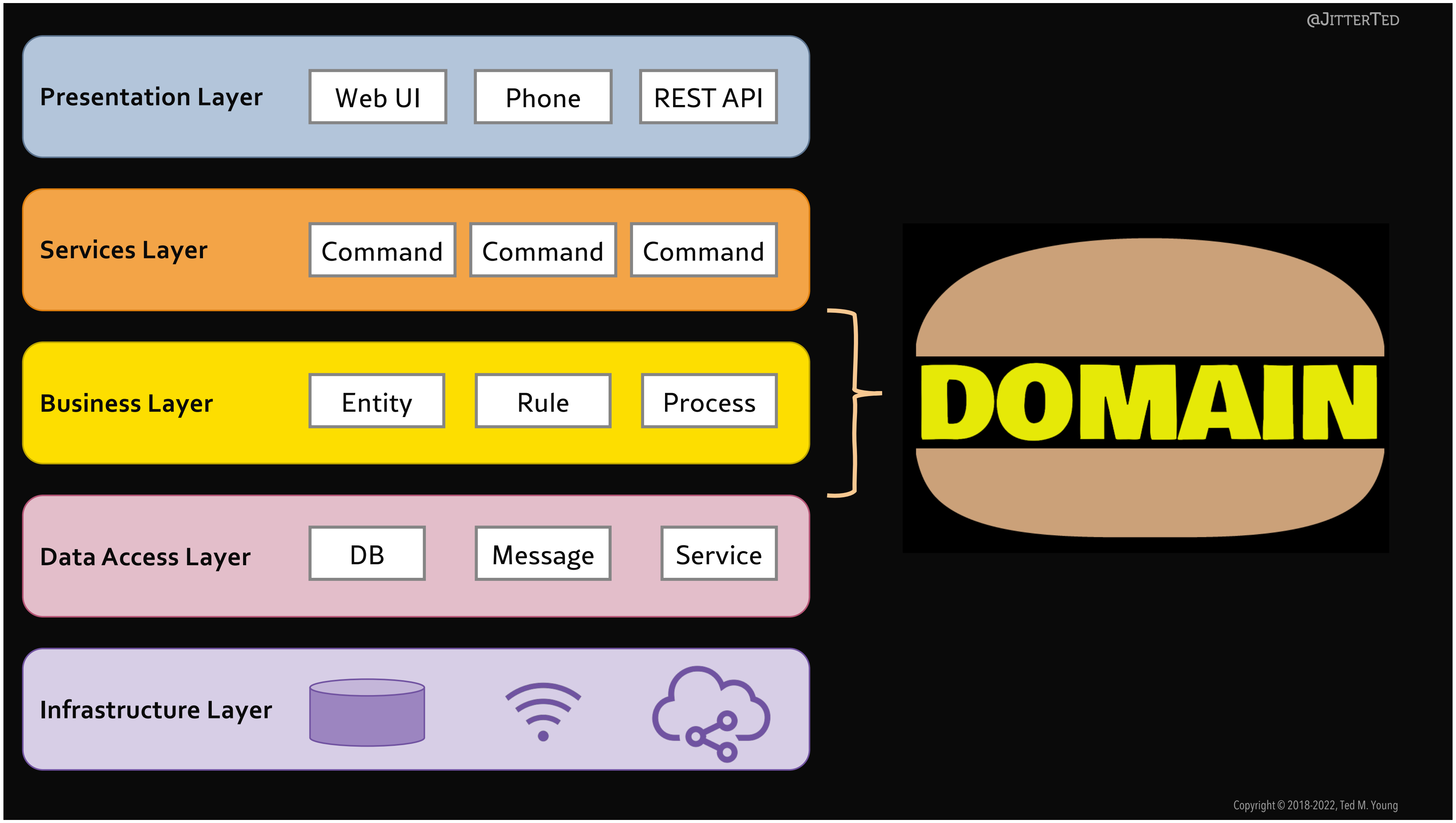
Vertically-Layered Architecture



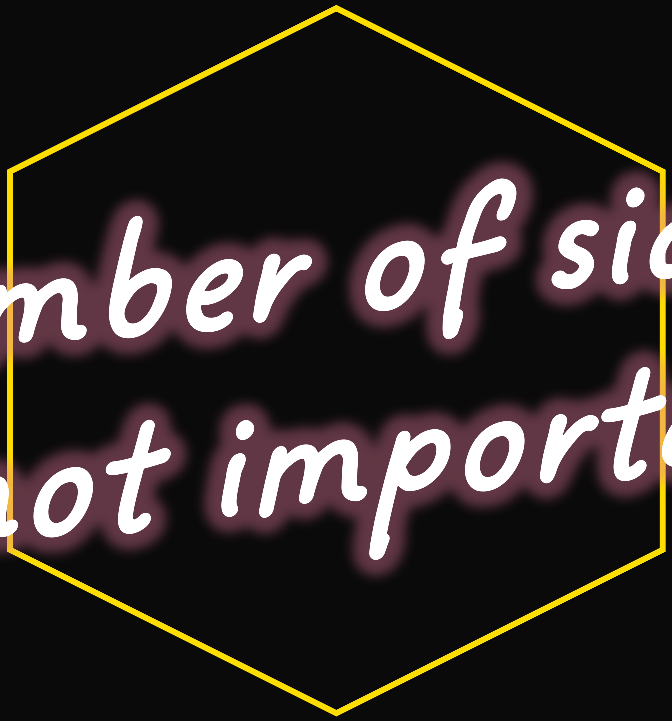
Which is the most important part?



This is what the business is paying for.


@JitterTed
Presentation Layer
Web UI
Phone
REST API
Services Layer
Command
Command
Command
Business Layer
Entity
Rule
Process
Data Access Layer
DB
Message
Service
Infrastructure Layer
DOMAIN

Hexagonal Architecture



Number of sides is not important

"Allow an application to equally be driven by users, programs, [and] automated tests... and to be developed and tested in isolation from its eventual run-time devices and databases."

Ports & Adapters Pattern
Alistair Cockburn

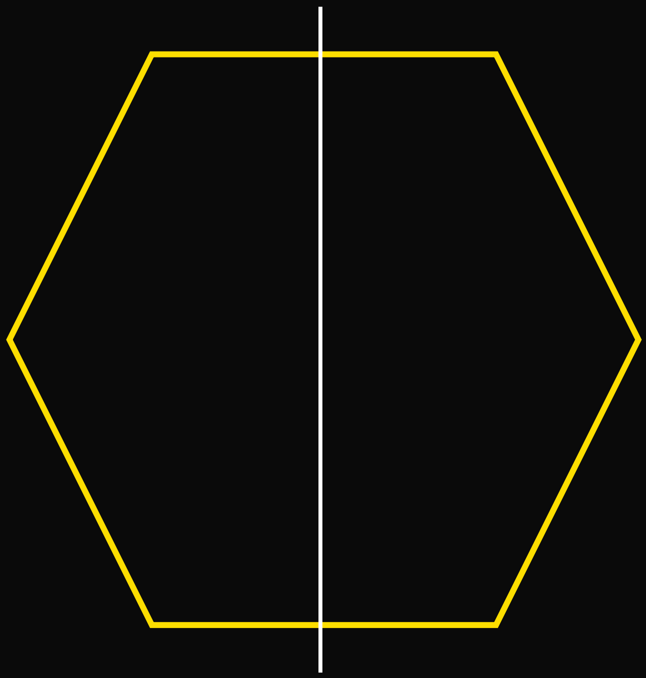
"Allow an application to equally be driven by users, programs, [and] automated tests… and to be developed and tested in isolation from its eventual run-time devices and databases."

Ports & Adapters Pattern
Alistair Cockburn

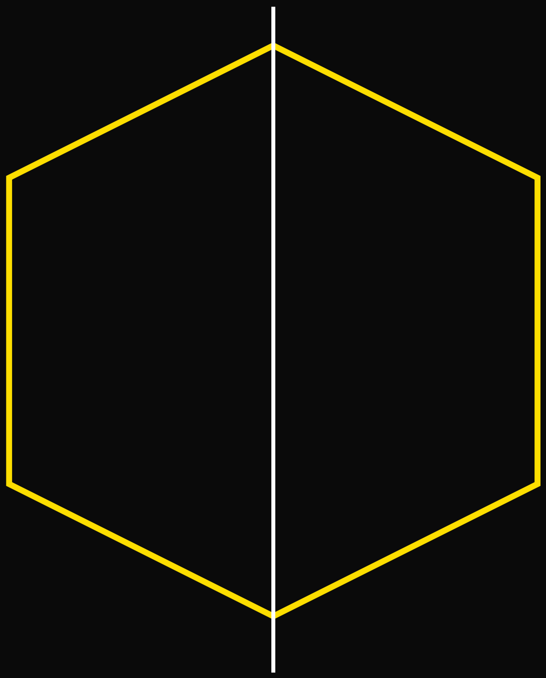
"Allow an application to equally be driven by users, programs, [and] automated tests... and to be developed and tested in isolation from its eventual run-time devices and databases."

Ports & Adapters Pattern
Alistair Cockburn

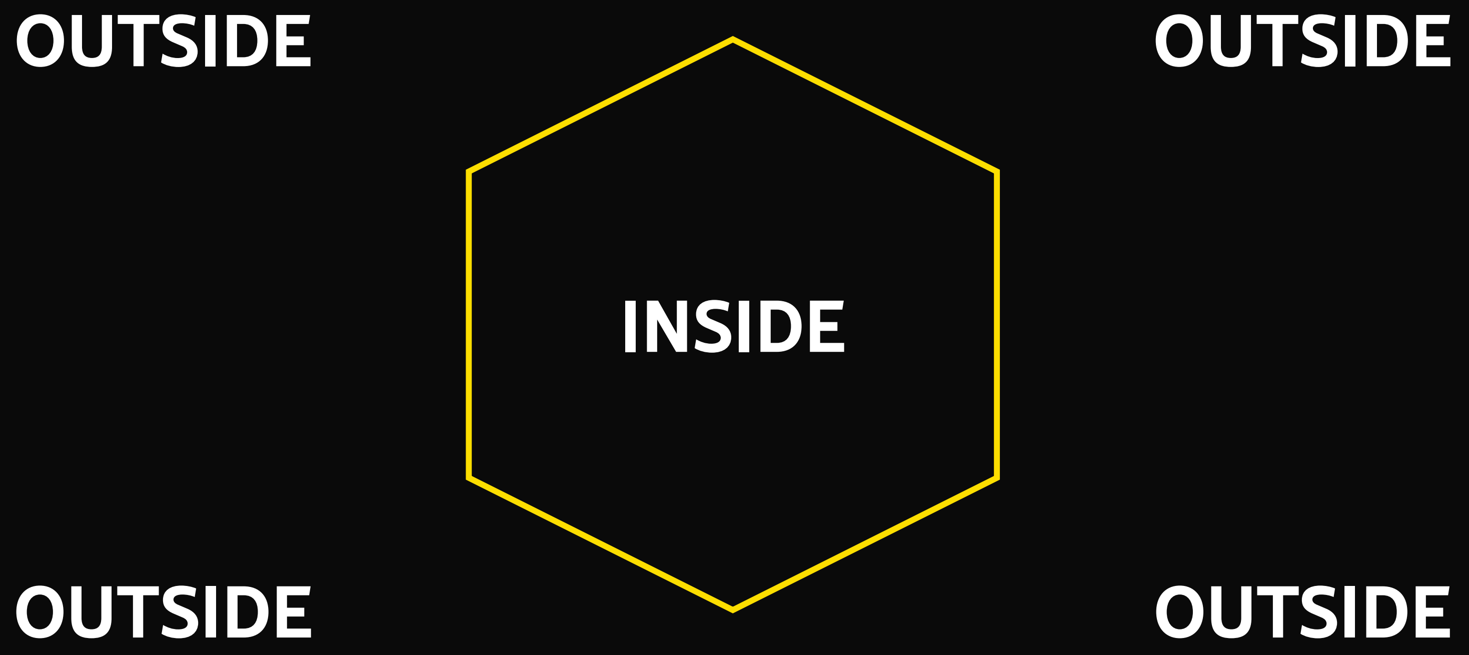
FLAT-TOPPED

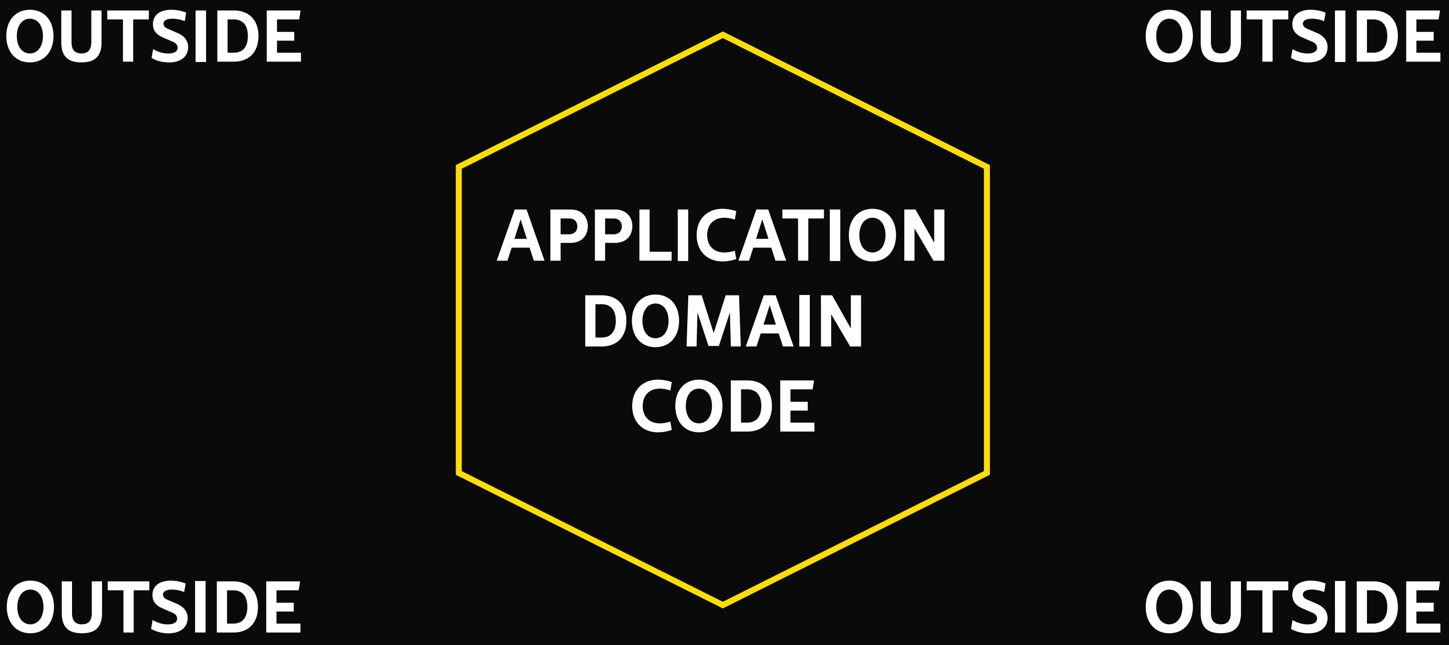


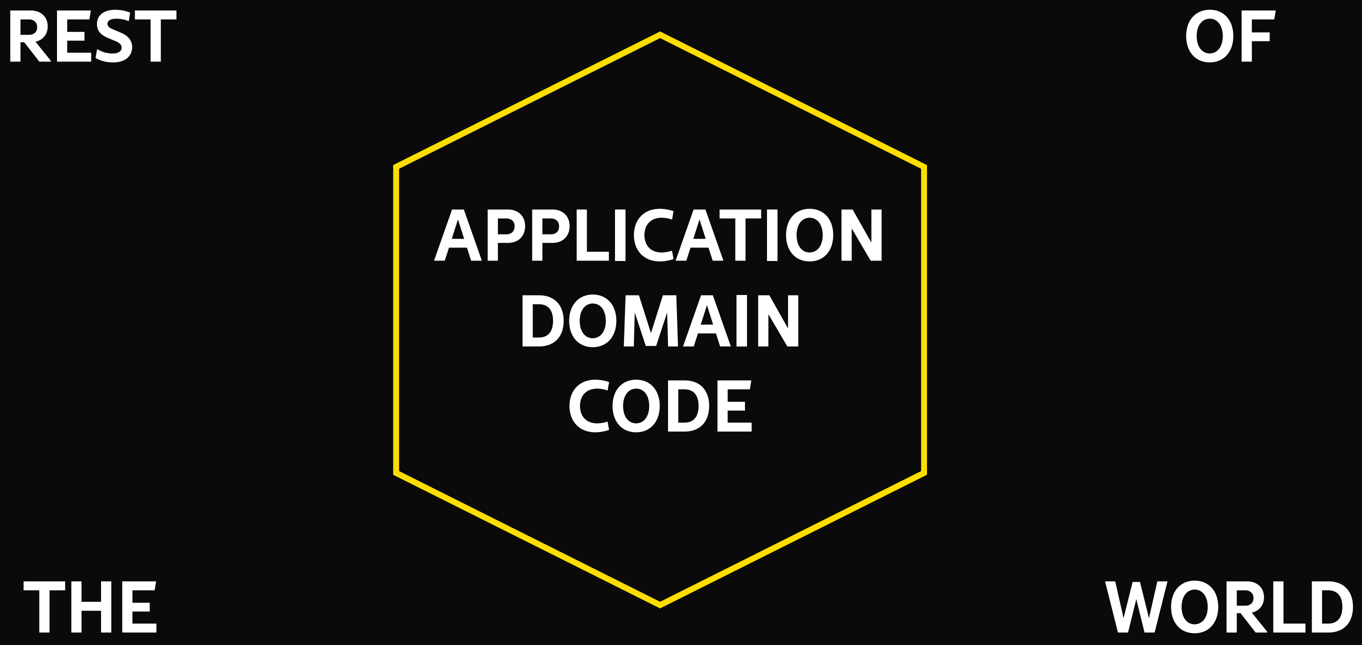
POINTY-TOPPED

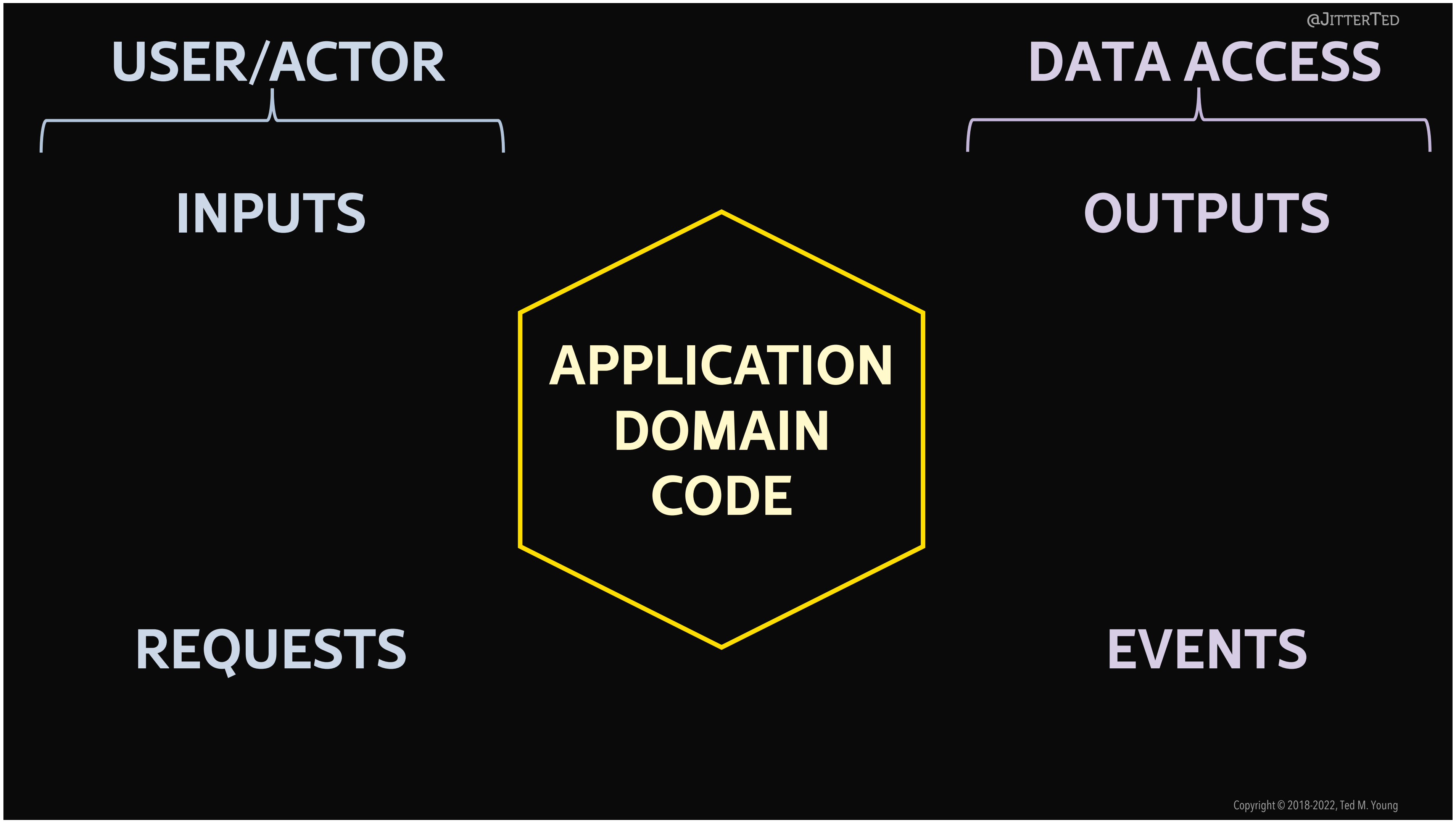


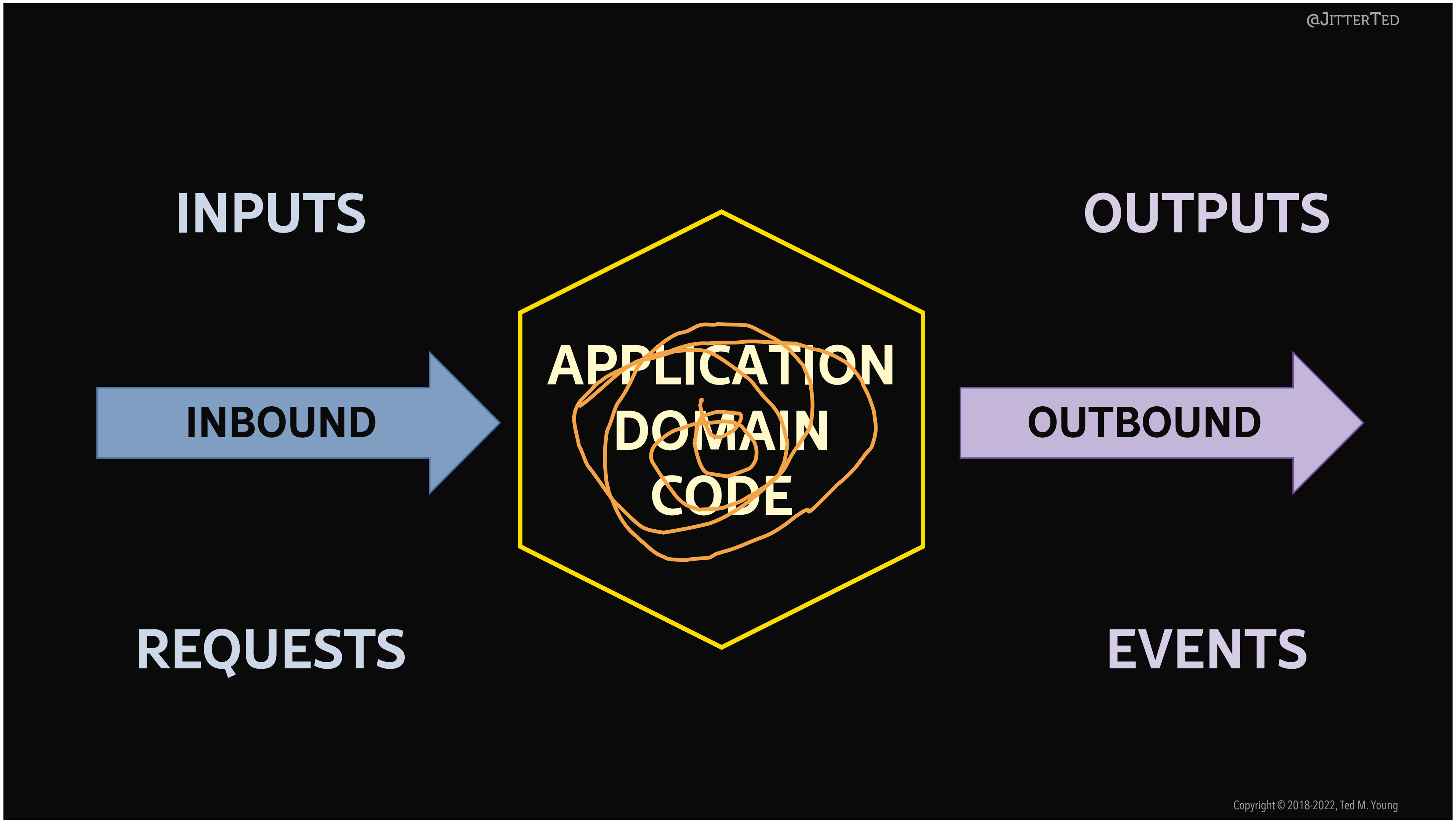
@JitterTed

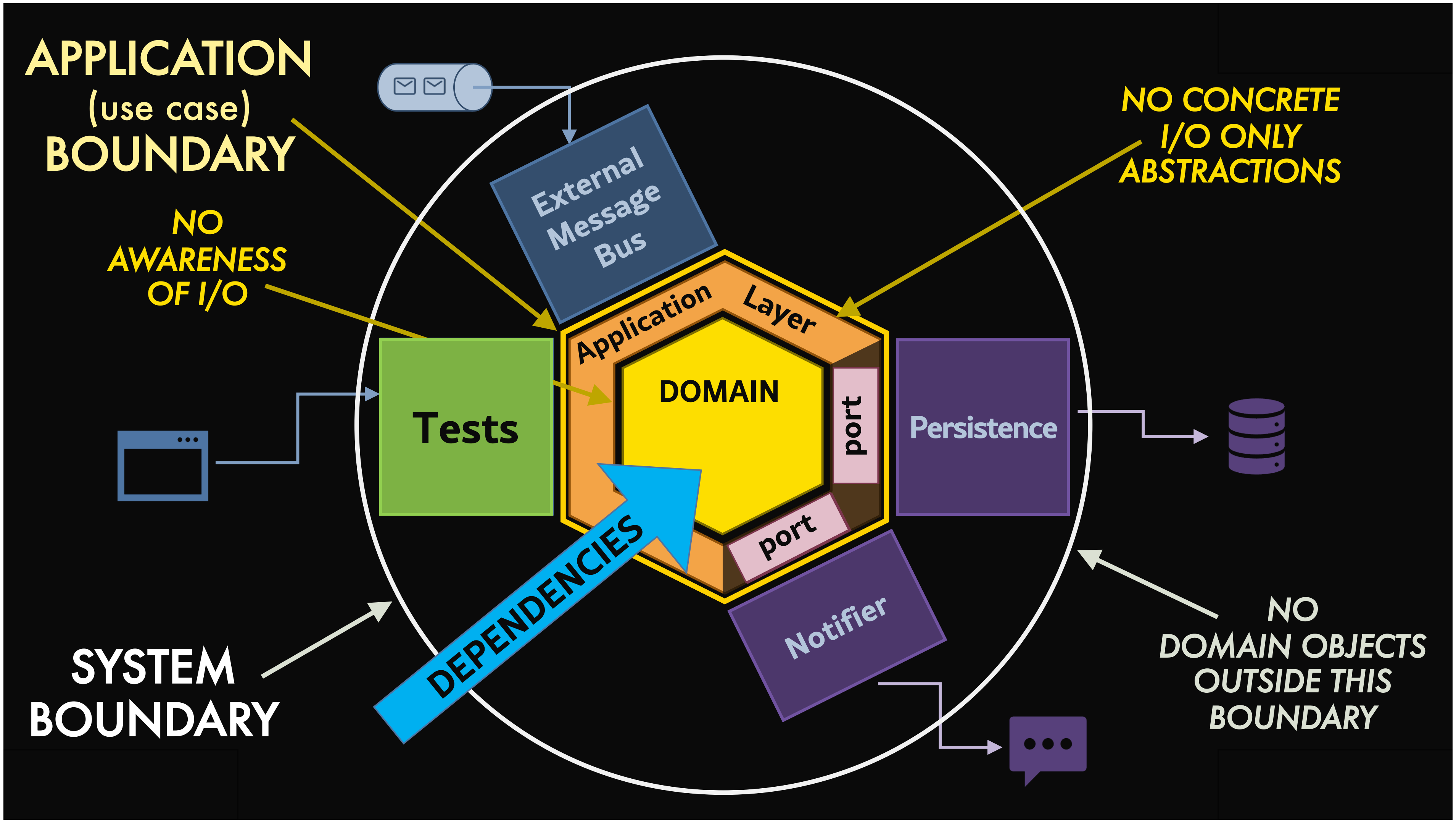



OUTSIDE
OUTSIDE
APPLICATION
DOMAIN
CODE
OUTSIDE
OUTSIDE


REST
OF
APPLICATION DOMAIN CODE
THE
WORLD


@JitterTed
USER/ACTOR
DATA ACCESS
INPUTS
OUTPUTS
APPLICATION DOMAIN CODE
REQUESTS
EVENTS

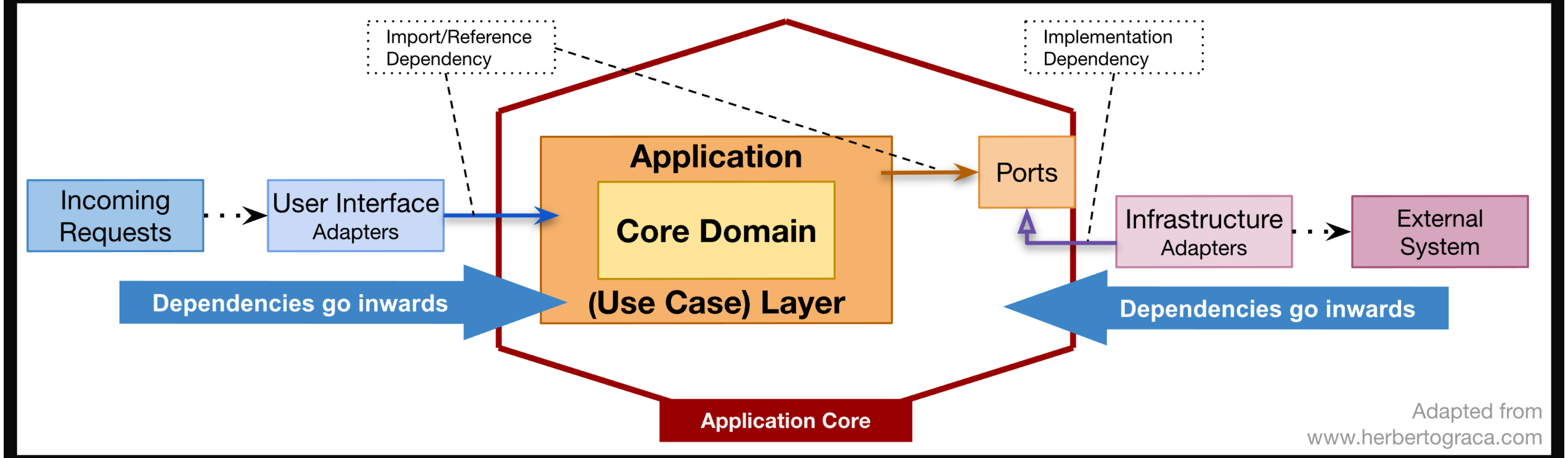

@JitterTed
INPUTS
OUTPUTS
INBOUND
APPLICATION DOMAIN CODE
OUTBOUND
REQUESTS
EVENTS


APPLICATION
(use case)
BOUNDARY
NO AWARENESS OF I/O
External Message Bus
NO CONCRETE I/O ONLY ABSTRACTIONS
Application Layer
DOMAIN
port
port
Tests
Persistence
Notifier
DEPENDENCIES
SYSTEM BOUNDARY
NO DOMAIN OBJECTS OUTSIDE THIS BOUNDARY

Dependency Rule

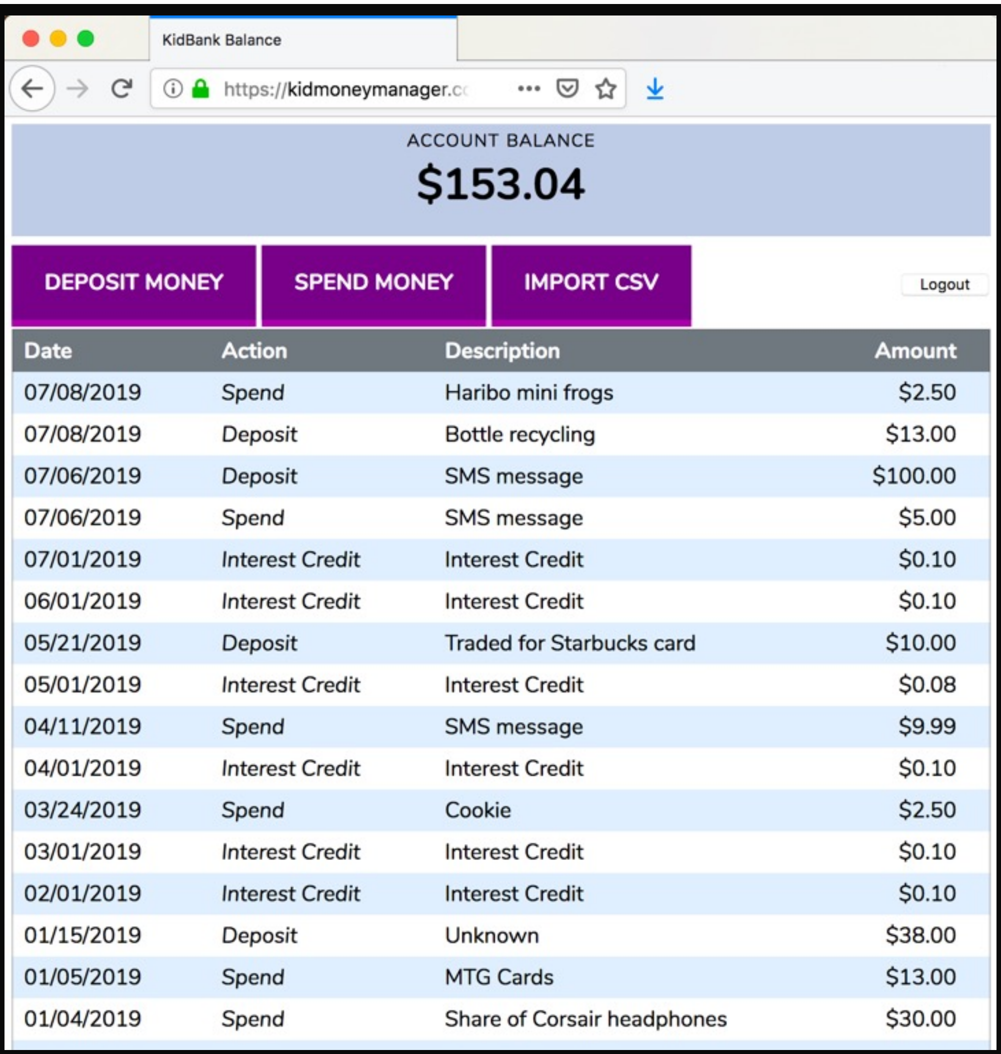
Dependencies Point Inwards

Dependencies Point Inward



Live Coding Recordings: https://JitterTed.TV

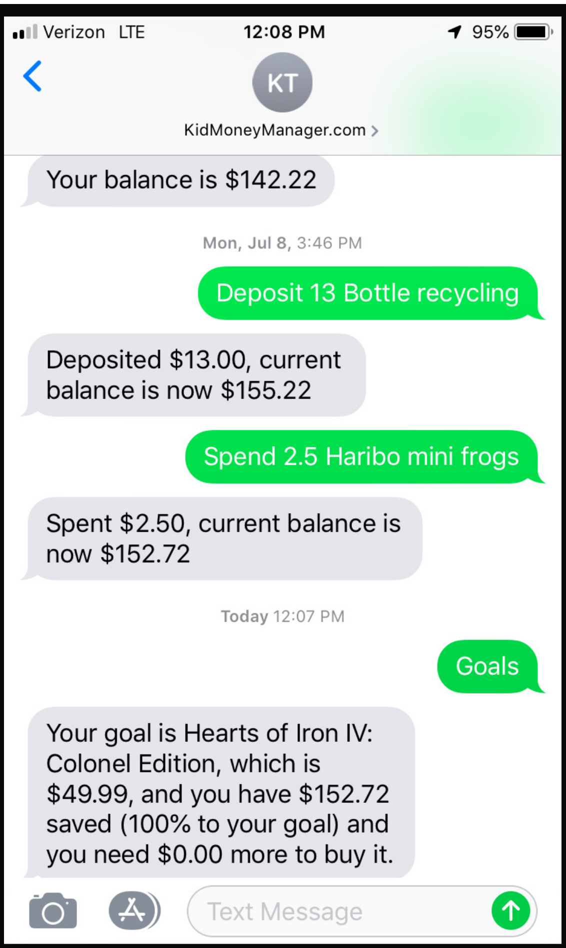
Source Code: https://github.com/tedyoung/kid-bank



ACCOUNT BALANCE

$153.04

Date	Action	Description	Amount
07/08/2019	Spend	Haribo mini frogs	$2.50
07/08/2019	Deposit	Bottle recycling	$13.00
07/06/2019	Deposit	SMS message	$100.00
07/06/2019	Spend	SMS message	$5.00
07/01/2019	Interest Credit	Interest Credit	$0.10
06/01/2019	Interest Credit	Interest Credit	$0.10
05/21/2019	Deposit	Traded for Starbucks card	$10.00
05/01/2019	Interest Credit	Interest Credit	$0.08
04/11/2019	Spend	SMS message	$9.99
04/01/2019	Interest Credit	Interest Credit	$0.10
03/24/2019	Spend	Cookie	$2.50
03/01/2019	Interest Credit	Interest Credit	$0.10
02/01/2019	Interest Credit	Interest Credit	$0.10
01/15/2019	Deposit	Unknown	$38.00
01/05/2019	Spend	MTG Cards	$13.00
01/04/2019	Spend	Share of Corsair headphones	$30.00



Your balance is $142.22

Mon, Jul 8, 3:46 PM

Deposit 13 Bottle recycling

Deposited $13.00, current balance is now $155.22

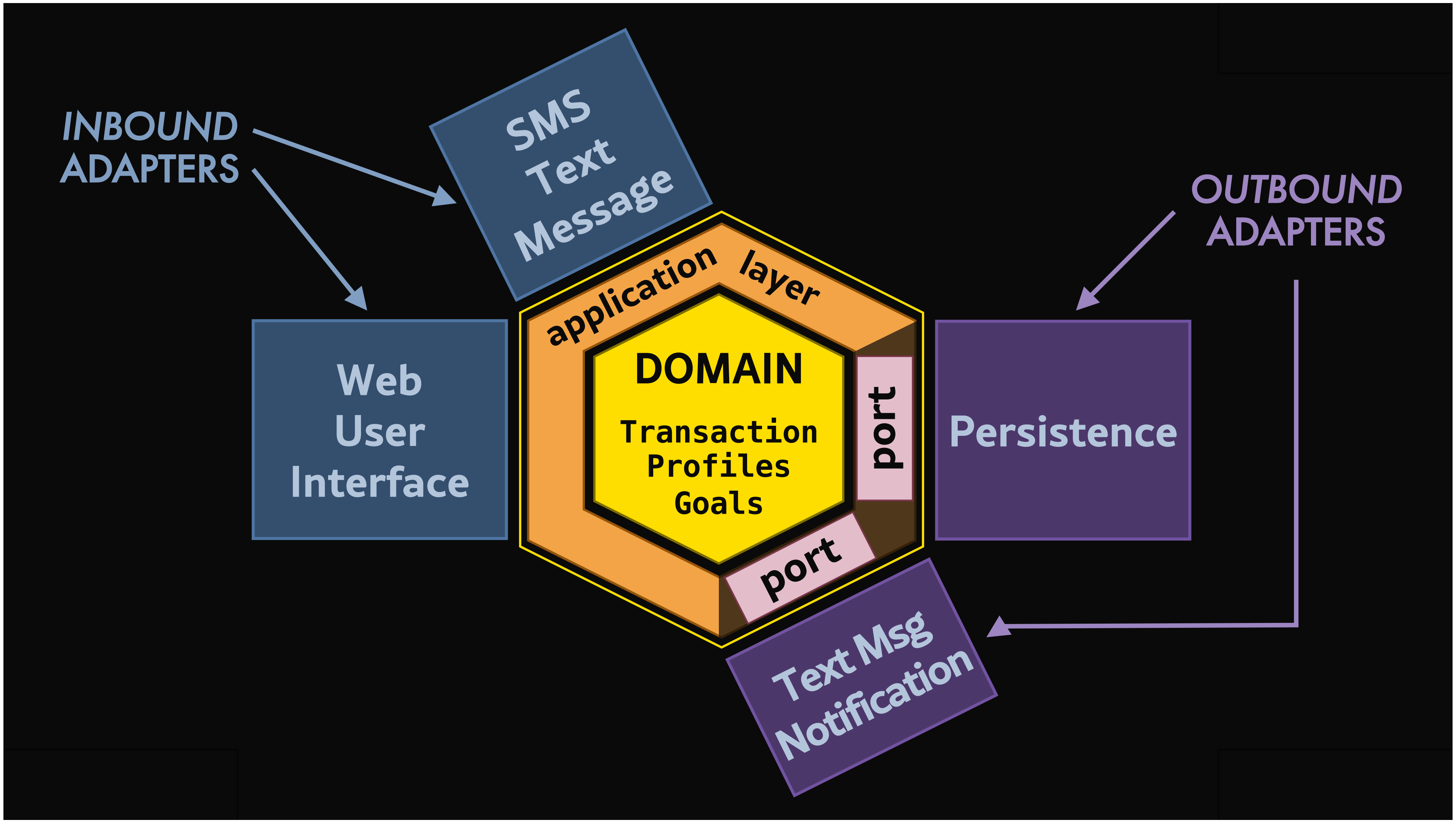
Spend 2.5 Haribo mini frogs

Spent $2.50, current balance is now $152.72

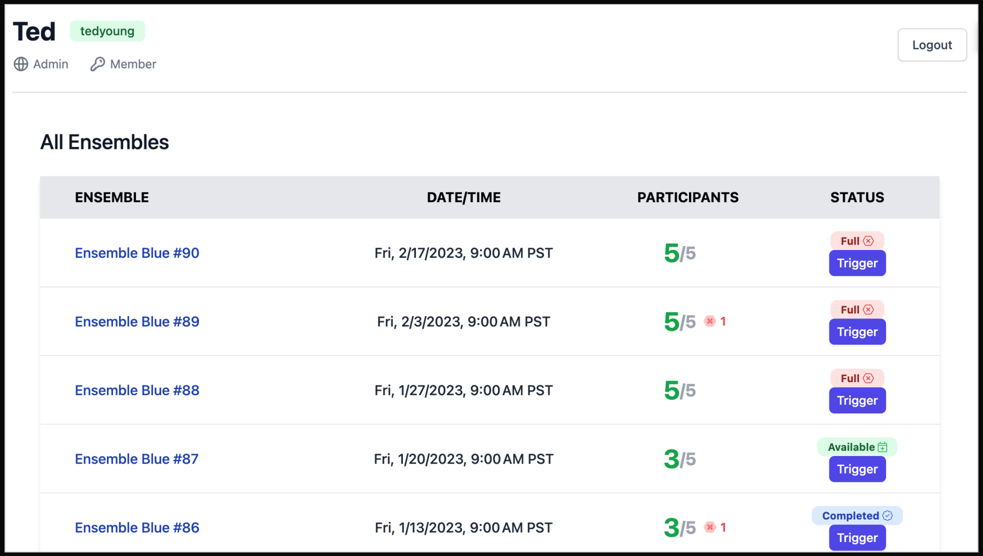
Today 12:07 PM

Goals

Your goal is Hearts of Iron IV: Colonel Edition, which is $49.99, and you have $152.72 saved (100% to your goal) and you need $0.00 more to buy it.


INBOUND ADAPTERS
SMS Text Message
Web User Interface
application layer
DOMAIN
Transaction
Profiles
Goals
port
port
OUTBOUND ADAPTERS
Persistence
Text Msg Notification

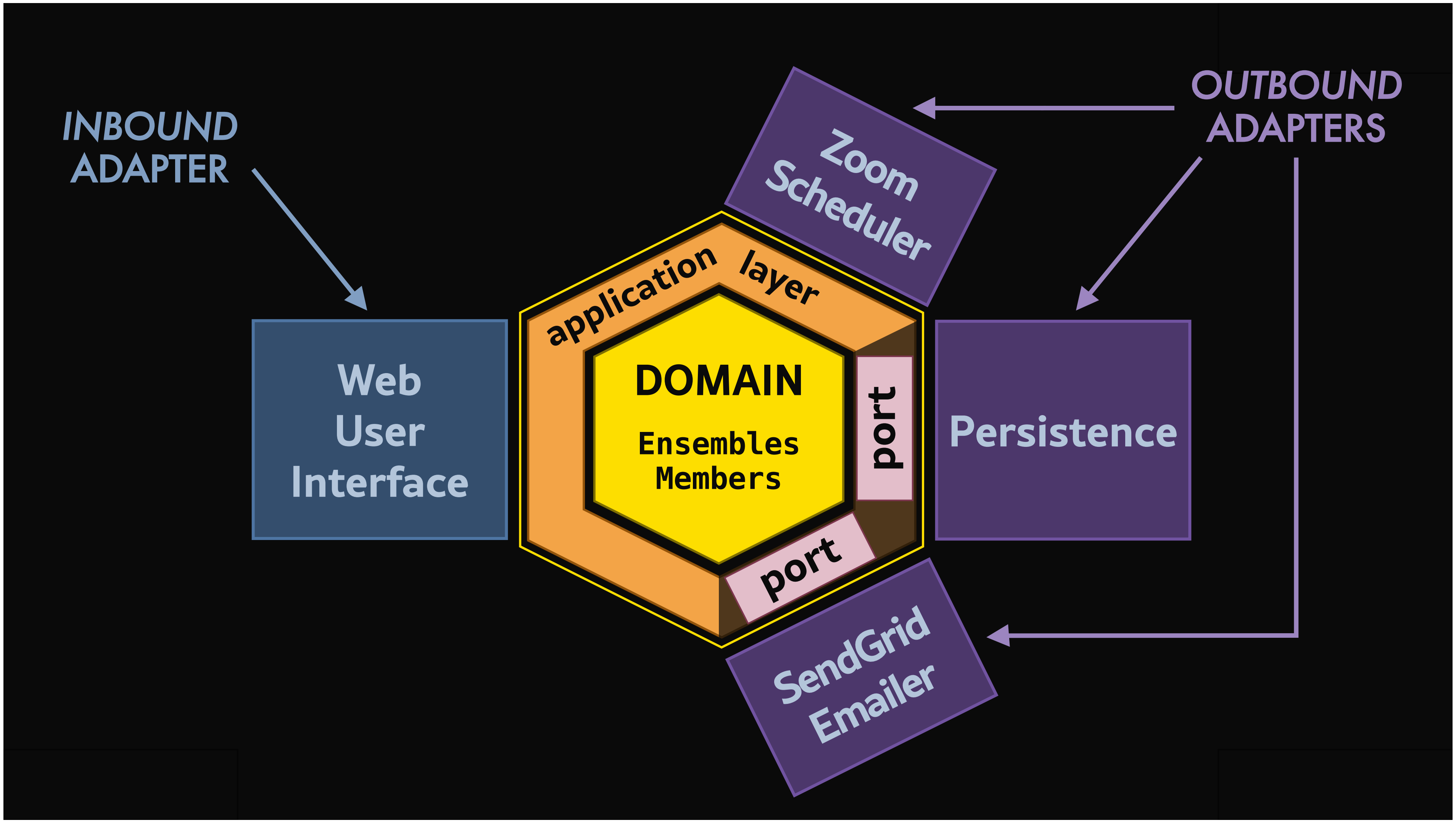
ENSEMBLER
managing ensemble events

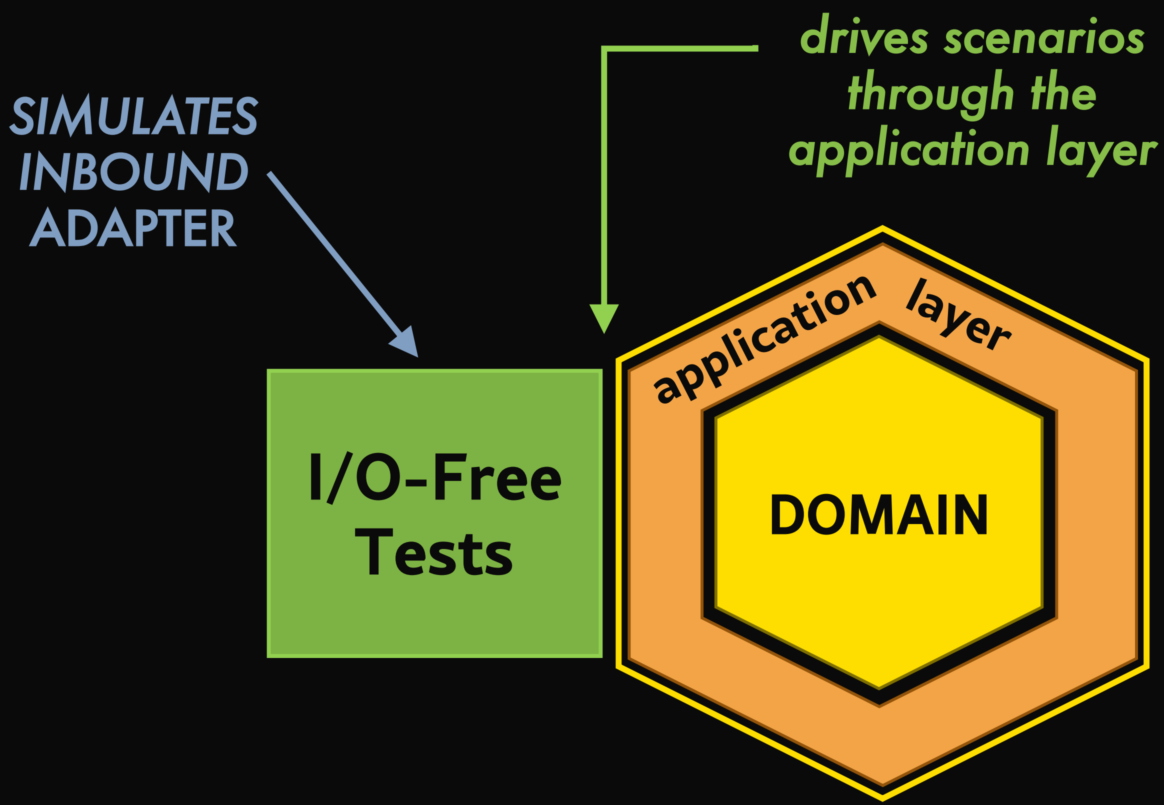

Ted tedyoung
Admin Member
Logout

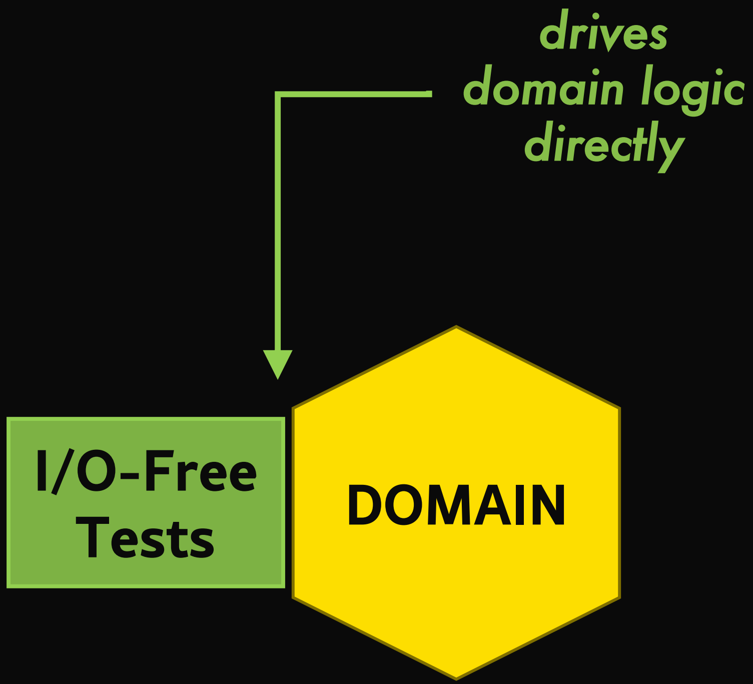
All Ensembles

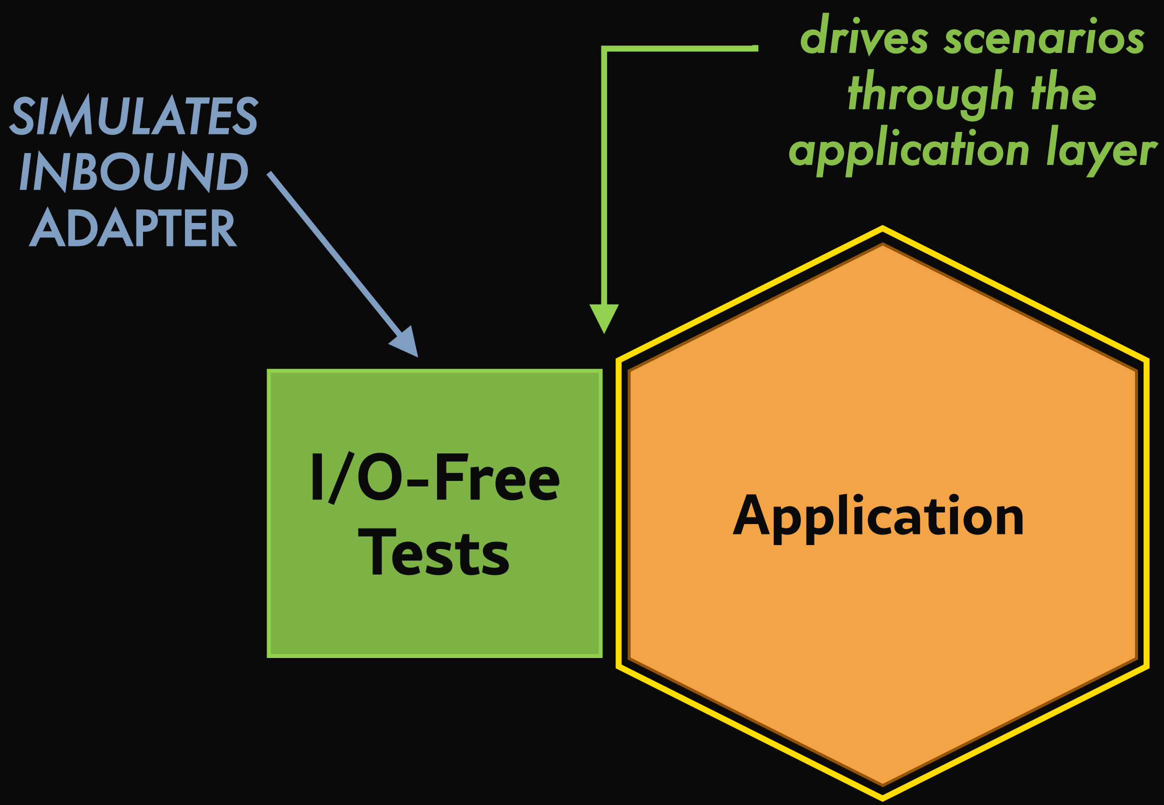
ENSEMBLE	DATE/TIME	PARTICIPANTS	STATUS
Ensemble Blue #90	Fri, 2/17/2023, 9:00 AM PST	5/5	Full Trigger
Ensemble Blue #89	Fri, 2/3/2023, 9:00 AM PST	5/5 ✗ 1	Full Trigger
Ensemble Blue #88	Fri, 1/27/2023, 9:00 AM PST	5/5	Full Trigger
Ensemble Blue #87	Fri, 1/20/2023, 9:00 AM PST	3/5	Available Trigger
Ensemble Blue #86	Fri, 1/13/2023, 9:00 AM PST	3/5 ✗ 1	Completed Trigger

Source Code: https://github.com/jitterted/ensembler


INBOUND ADAPTER
OUTBOUND ADAPTERS
Web User Interface
Zoom Scheduler
application layer
DOMAIN
Ensembles
Members
port
port
Persistence
SendGrid Emailer

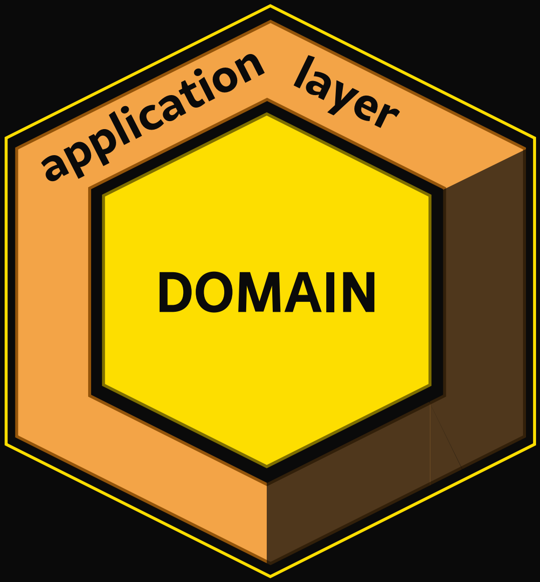

SIMULATES INBOUND ADAPTER
drives scenarios through the application layer
I/O-Free Tests
application layer
DOMAIN

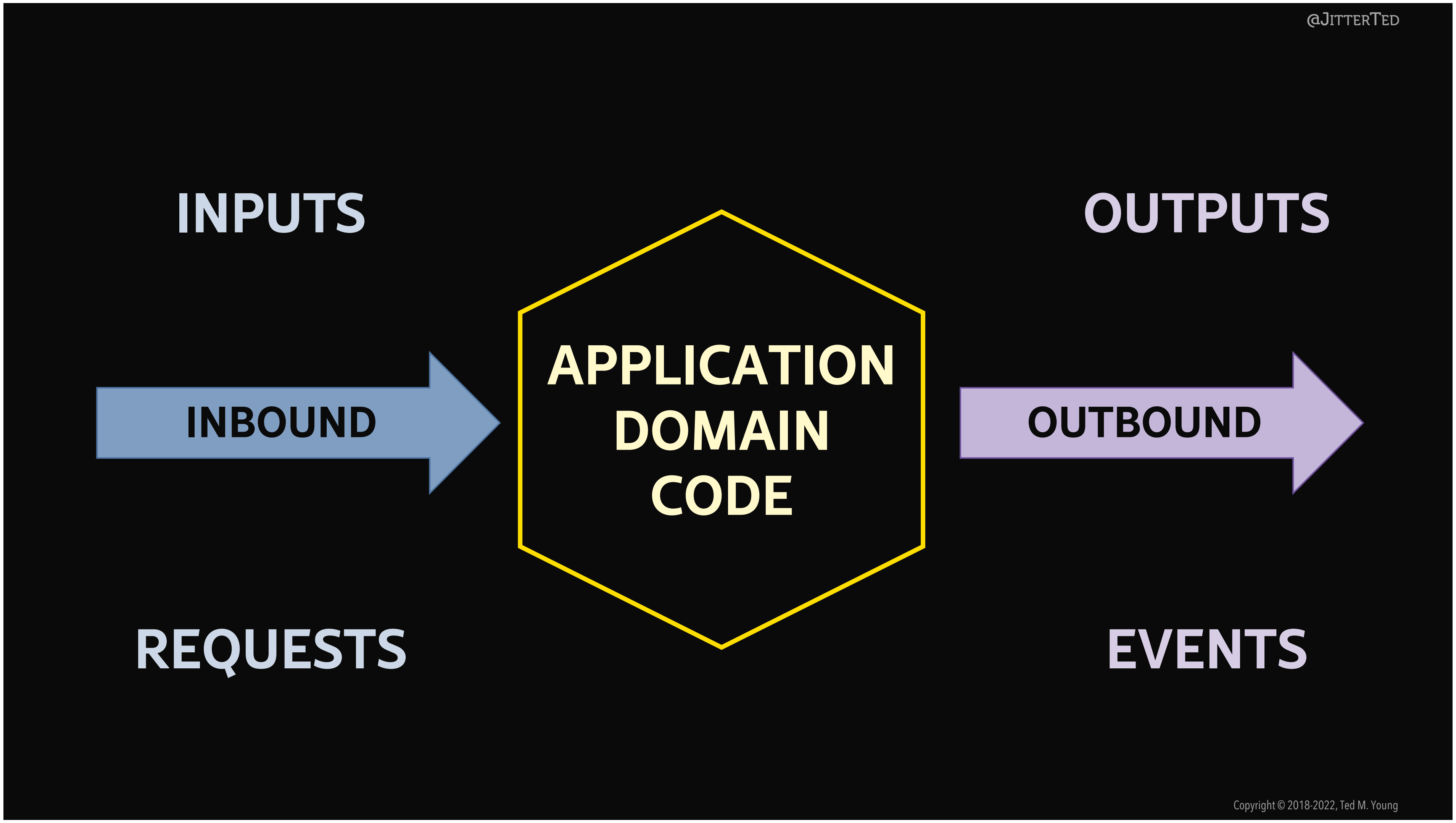

drives domain logic directly
I/O-Free Tests
DOMAIN

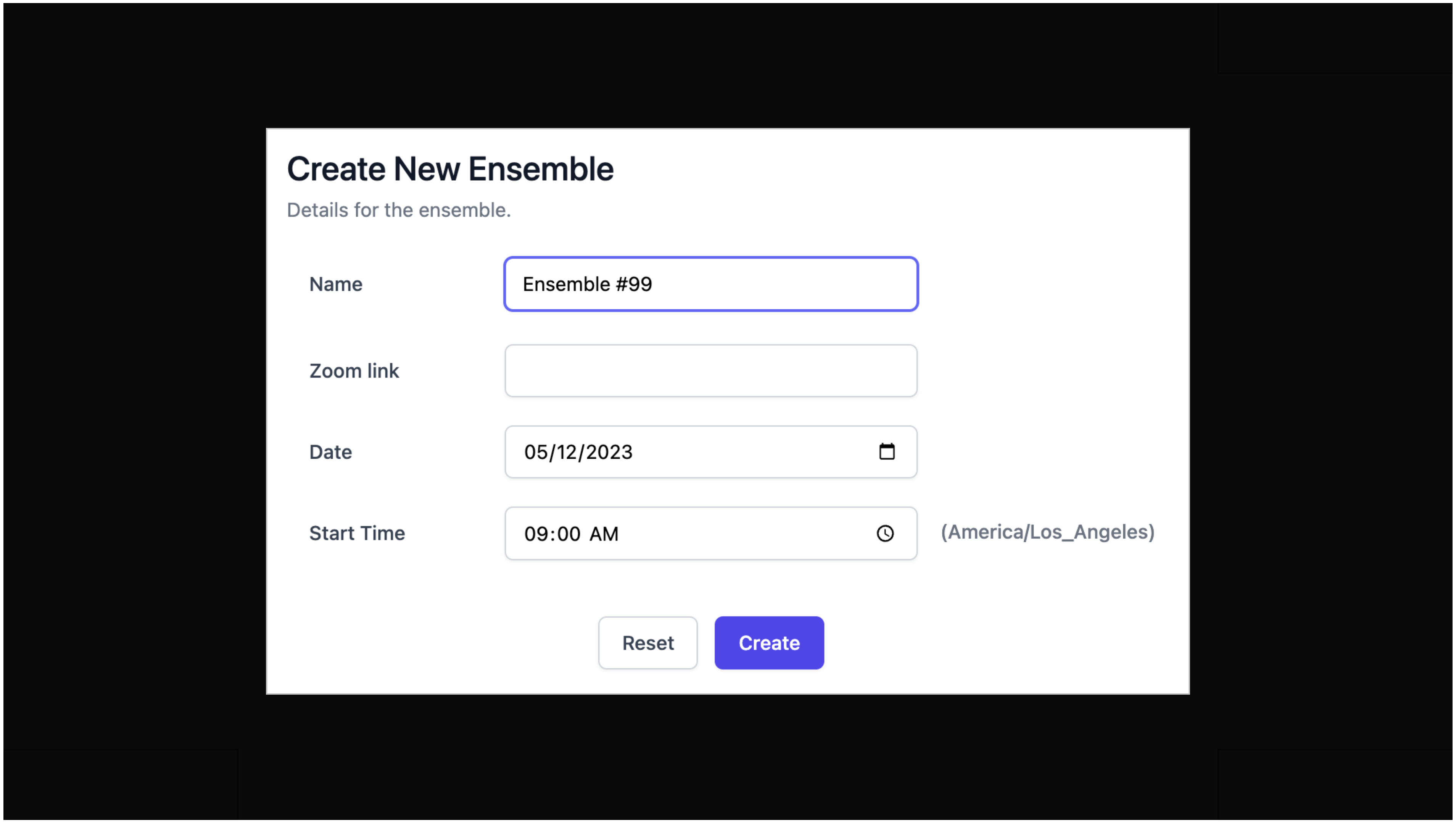

SIMULATES INBOUND ADAPTER
drives scenarios through the application layer
I/O-Free Tests
Application

Adapters

DOMAIN


application layer
DOMAIN

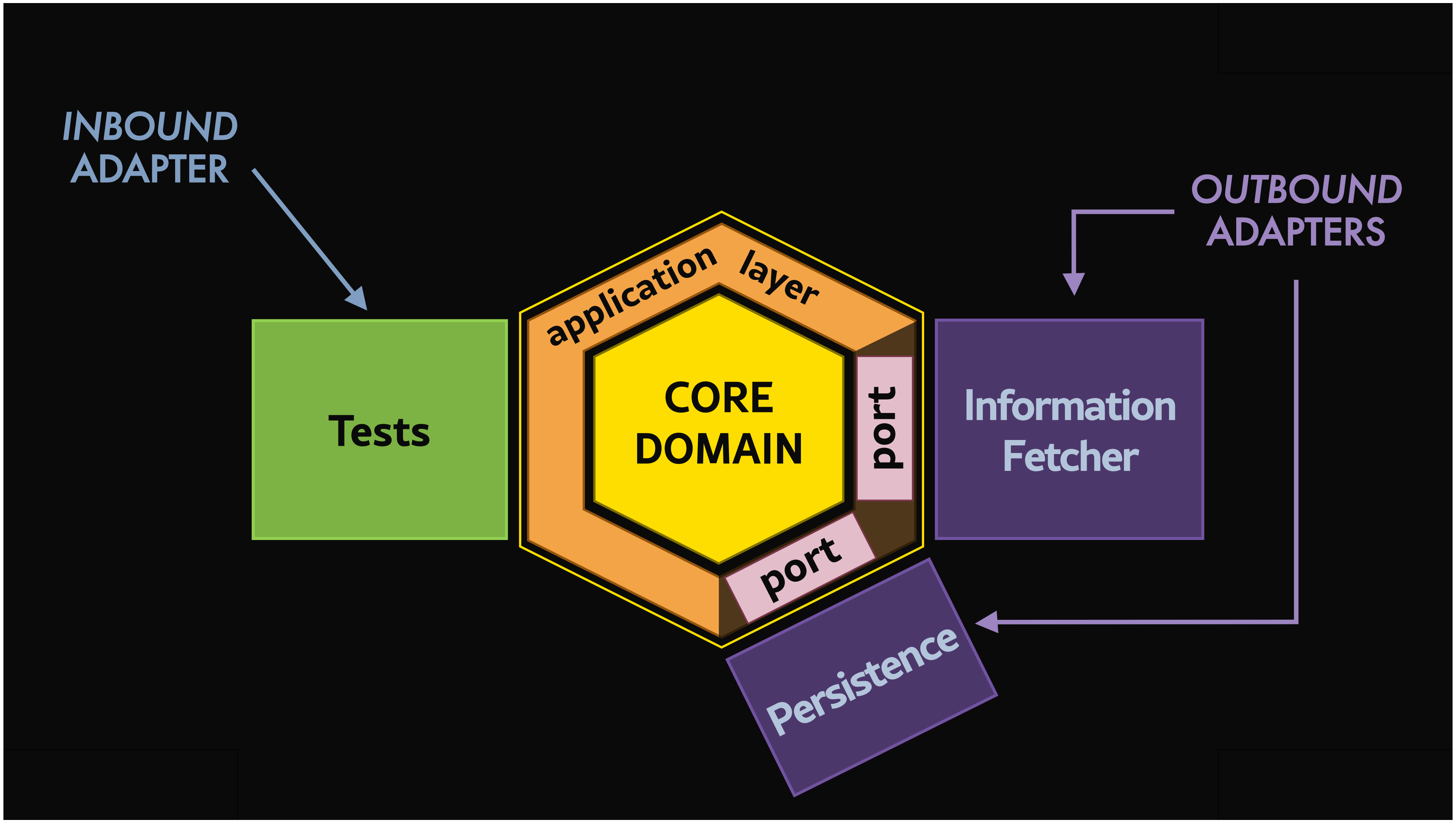

@JitterTed
INPUTS
OUTPUTS
INBOUND
APPLICATION DOMAIN CODE
OUTBOUND
REQUESTS
EVENTS

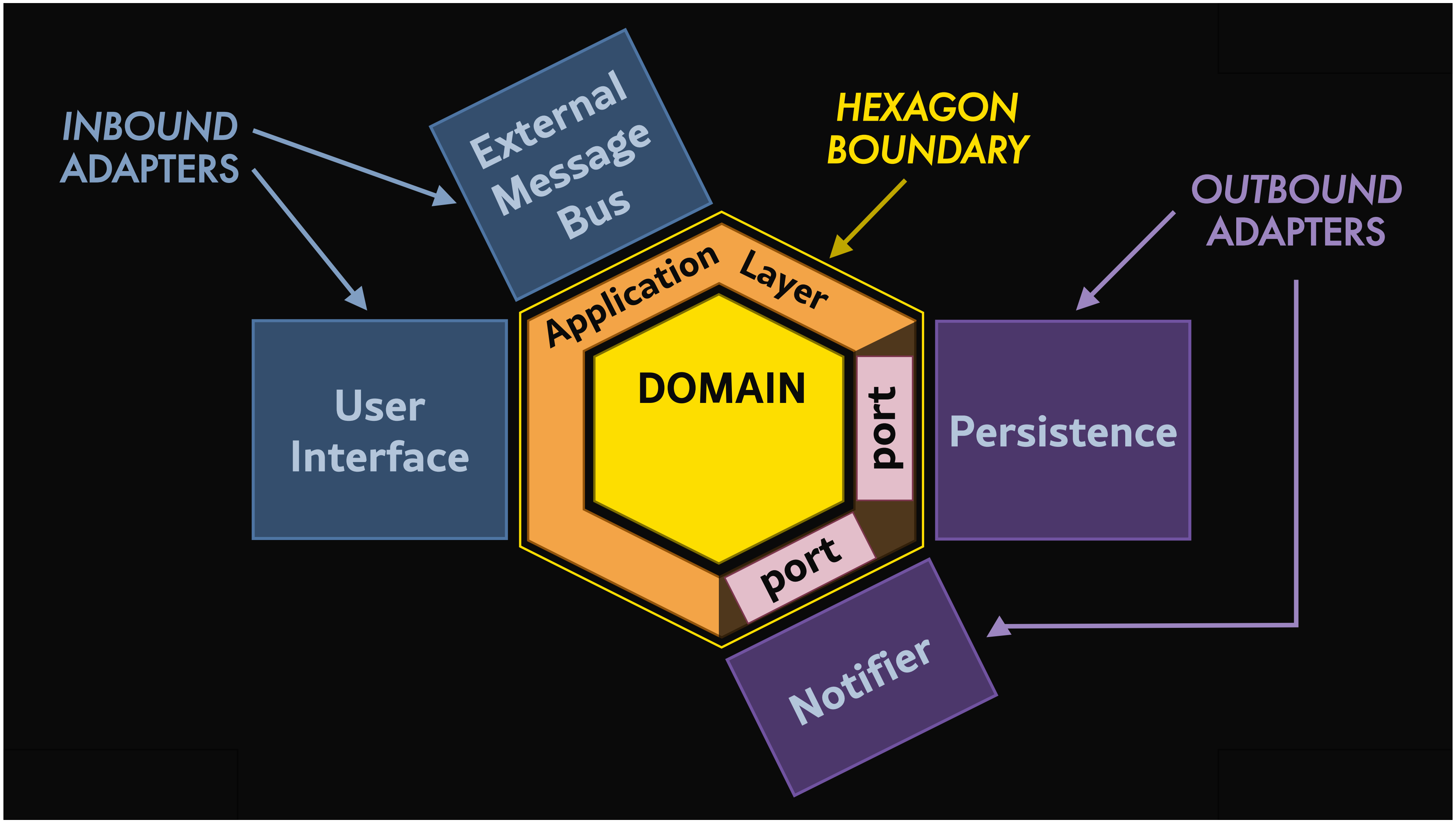


Create New Ensemble

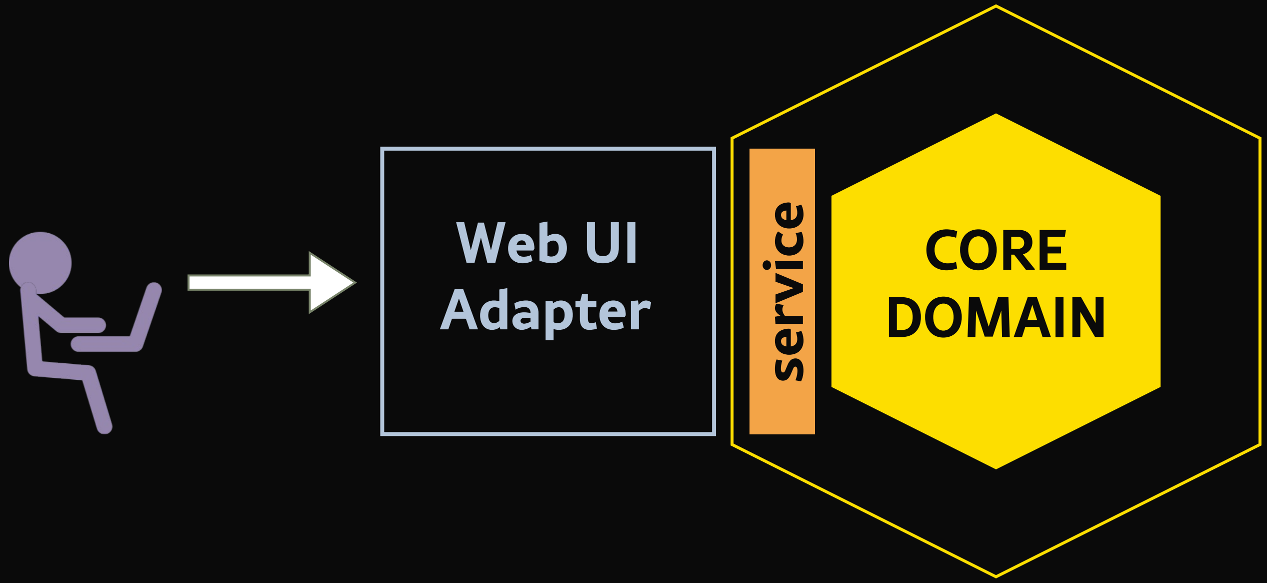
Details for the ensemble.

Name	Ensemble #99
Zoom link	
Date	05/12/2023
Start Time	09:00 AM (America/Los_Angeles)


INBOUND ADAPTER
OUTBOUND ADAPTERS
Tests
application layer
CORE DOMAIN
port
port
Information Fetcher
Persistence


INBOUND ADAPTERS
External Message Bus
HEXAGON BOUNDARY
OUTBOUND ADAPTERS
Application Layer
User Interface
DOMAIN
port
Persistence
port
Notifier

[Outside World]
User/Event-Driven



Domain Object

```
public class Account {

  public void deposit(LocalDateTime txnDate, int amount, String descr){...}

  public void spend(LocalDateTime txnDate, int amount, String descr) {...}

  public int balance() {...}

}
```

Domain Object

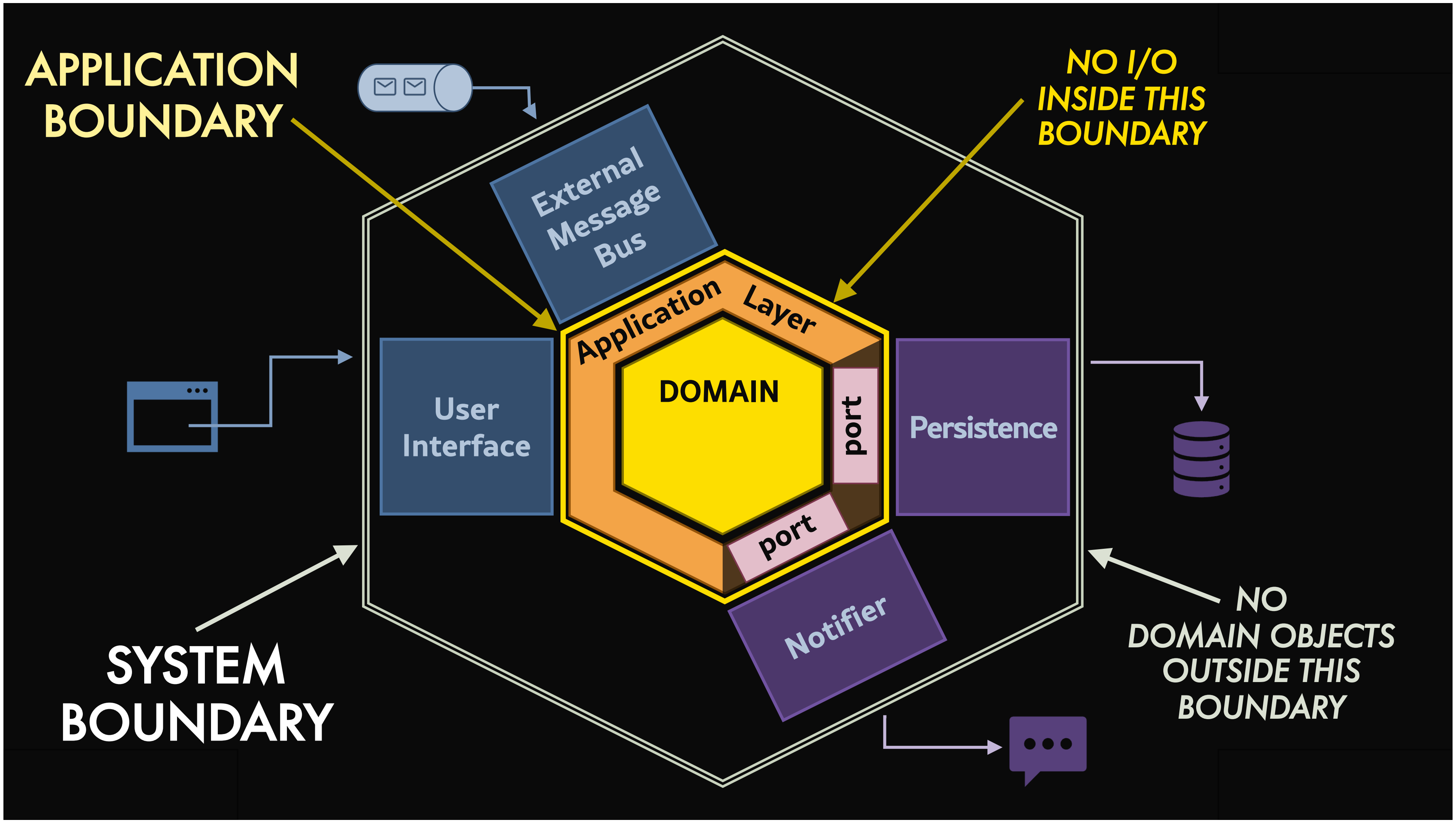
- Has Domain behavior
- Commands & Queries
  - get/set *only* allowed for data not owned by object
    - *Extrinsic* data, e.g.:
      - IDs
      - Audit timestamps

Data Transfer Object

- Only Data
  - Provided through getter/setters
  - Are JavaBeans
  - Have no behavior
    - Except for transformation from Domain, as a static method

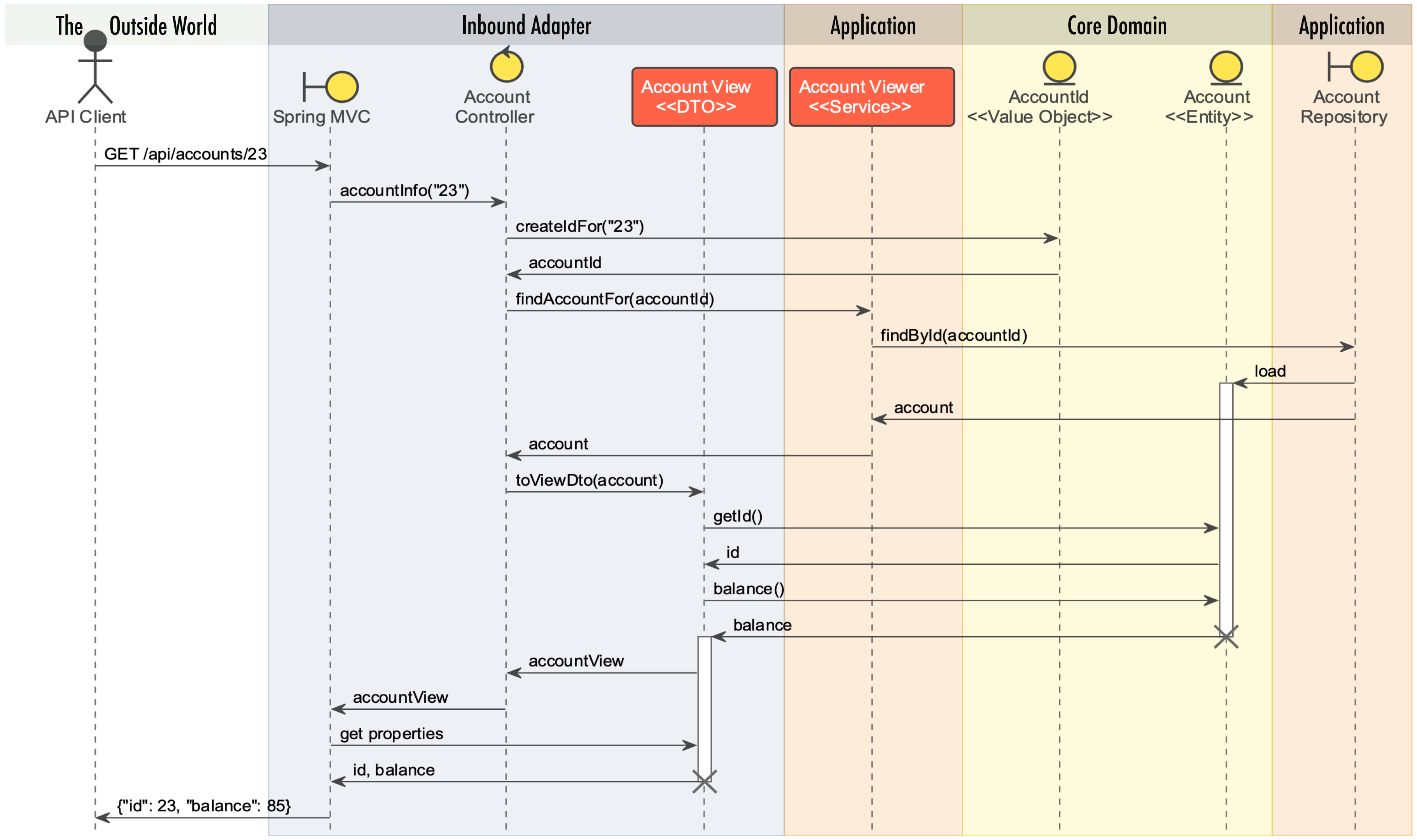
Rule

Domain Objects Never Leave the Adapter


APPLICATION BOUNDARY
NO I/O INSIDE THIS BOUNDARY
External Message Bus
Application Layer
DOMAIN
port
port
User Interface
Persistence
Notifier
SYSTEM BOUNDARY
NO DOMAIN OBJECTS OUTSIDE THIS BOUNDARY

Goal

Reduce Coupling


The Outside World
Inbound Adapter
Application
Core Domain
Application
API Client
Spring MVC
Account Controller
Account View <<DTO>>
Account Viewer <<Service>>
AccountId <<Value Object>>
Account <<Entity>>
Account Repository
GET /api/accounts/23
accountInfo("23")
createIdFor("23")
accountId
findAccountFor(accountId)
findById(accountId)
load
account
account
toViewDto(account)
getId()
id
balance()
balance
accountView
accountView
get properties
id, balance
{"id": 23, "balance": 85}

Adapter Role

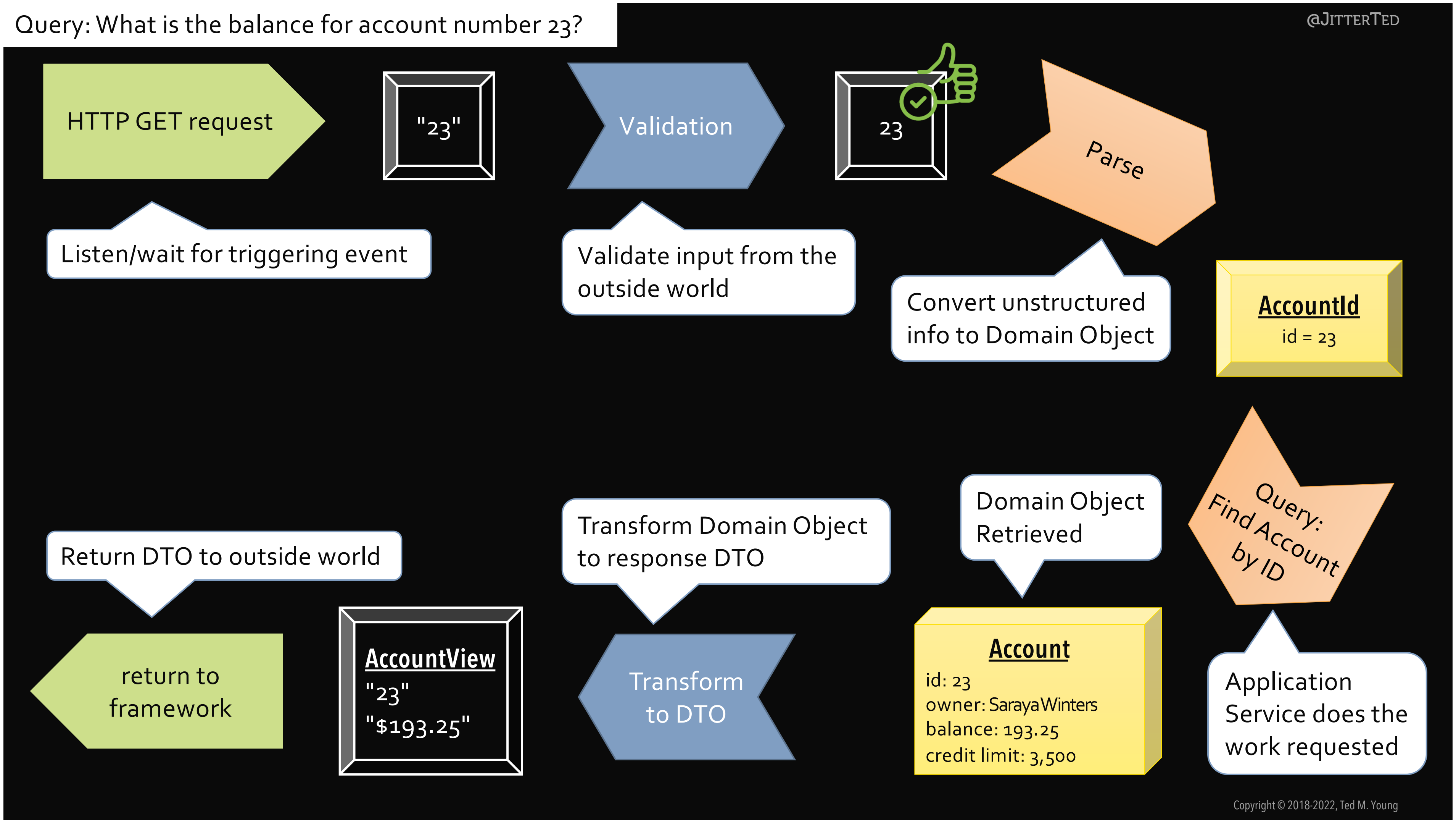
Package of Classes

Adapter Purpose

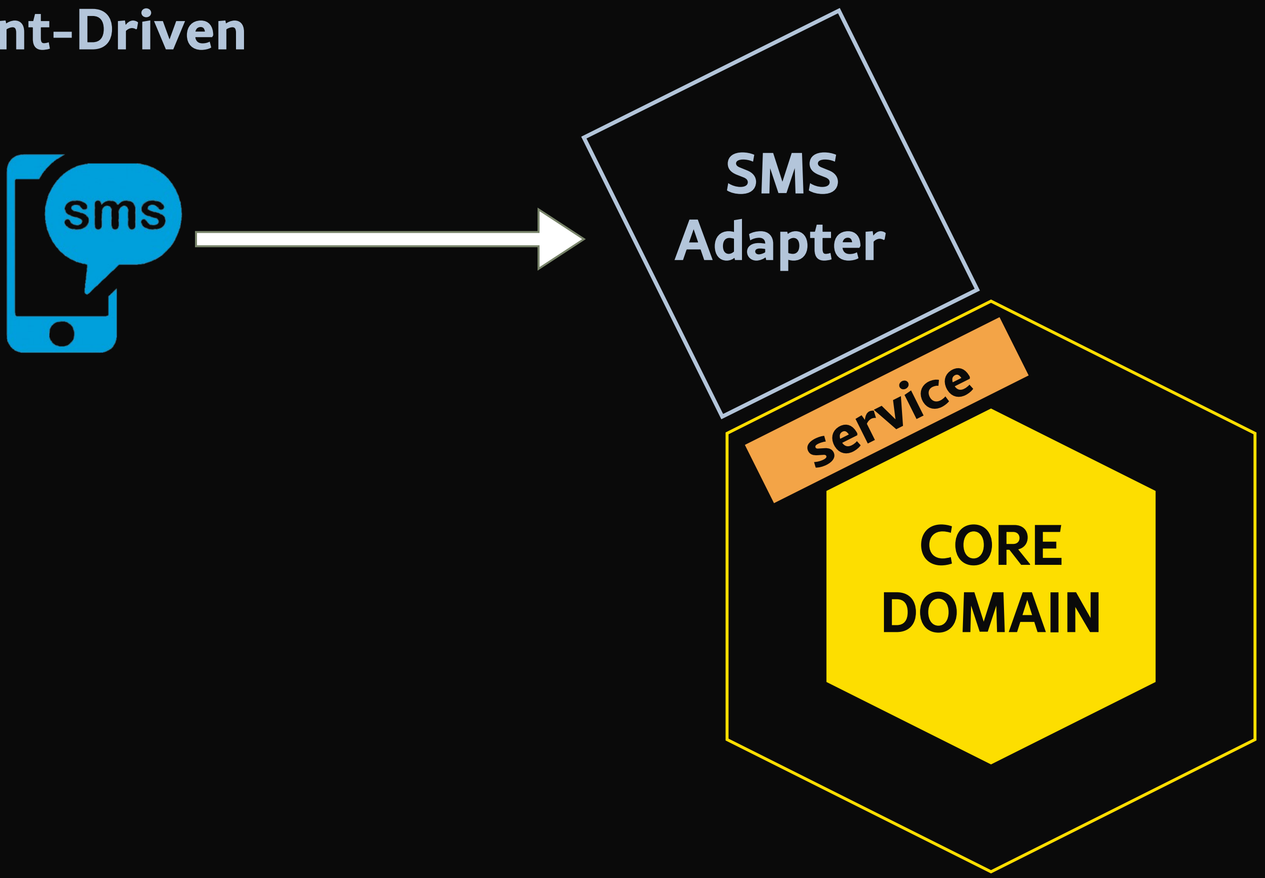
Translate/Transform

Outside World ↔️ Domain

Query: What is the balance for account number 23?


@JitterTed
HTTP GET request
"23"
Validation
23
Parse
Listen/wait for triggering event
Validate input from the outside world
Convert unstructured info to Domain Object
AccountId
id = 23
Query: Find Account by ID
Domain Object Retrieved
Transform Domain Object to response DTO
Return DTO to outside world
return to framework
AccountView
"23"
"$193.25"
Transform to DTO
Account
id: 23
owner: Saraya Winters
balance: 193.25
credit limit: 3,500
Application Service does the work requested

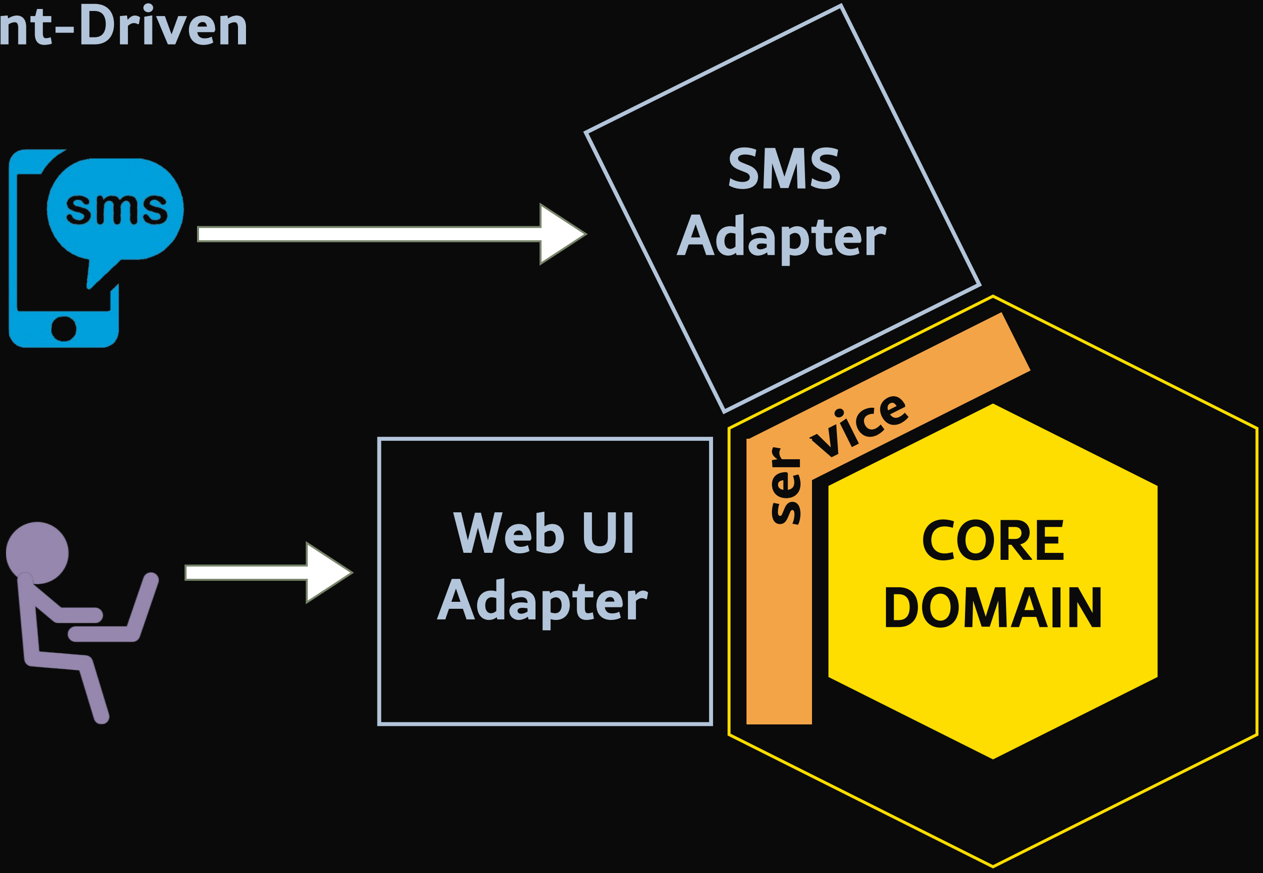
[Outside World]
User/Event-Driven



Rule

Adapter Per Communication Mechanism

[Outside World]
User/Event-Driven

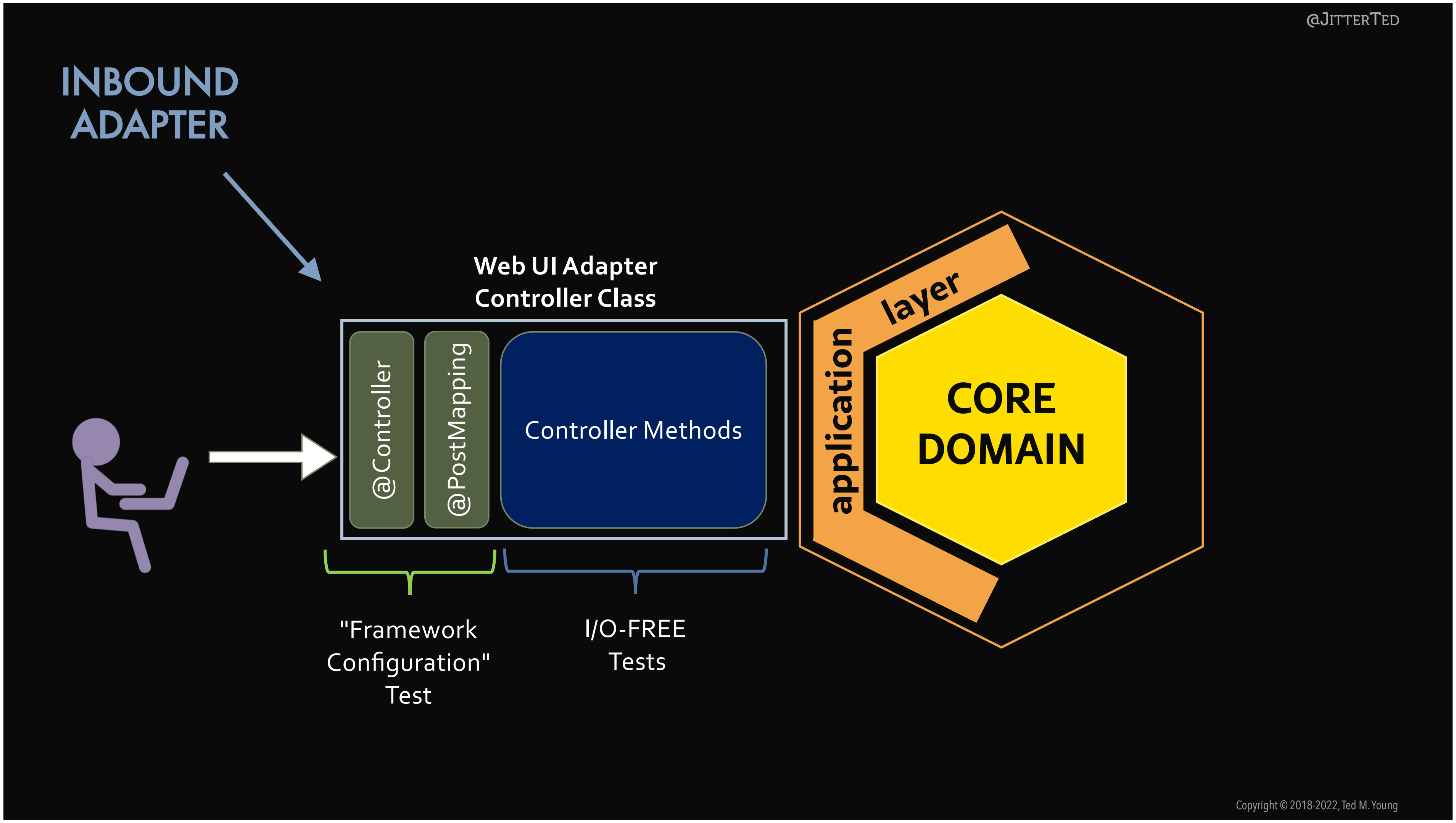


Dependency Rule

Adapters Don't Depend on Other Adapters

Adapter Testing

I/O and I/O-Free Testing


@JitterTed
INBOUND ADAPTER
Web UI Adapter Controller Class
@Controller
@PostMapping
Controller Methods
application layer
CORE DOMAIN
"Framework Configuration" Test
I/O-FREE Tests

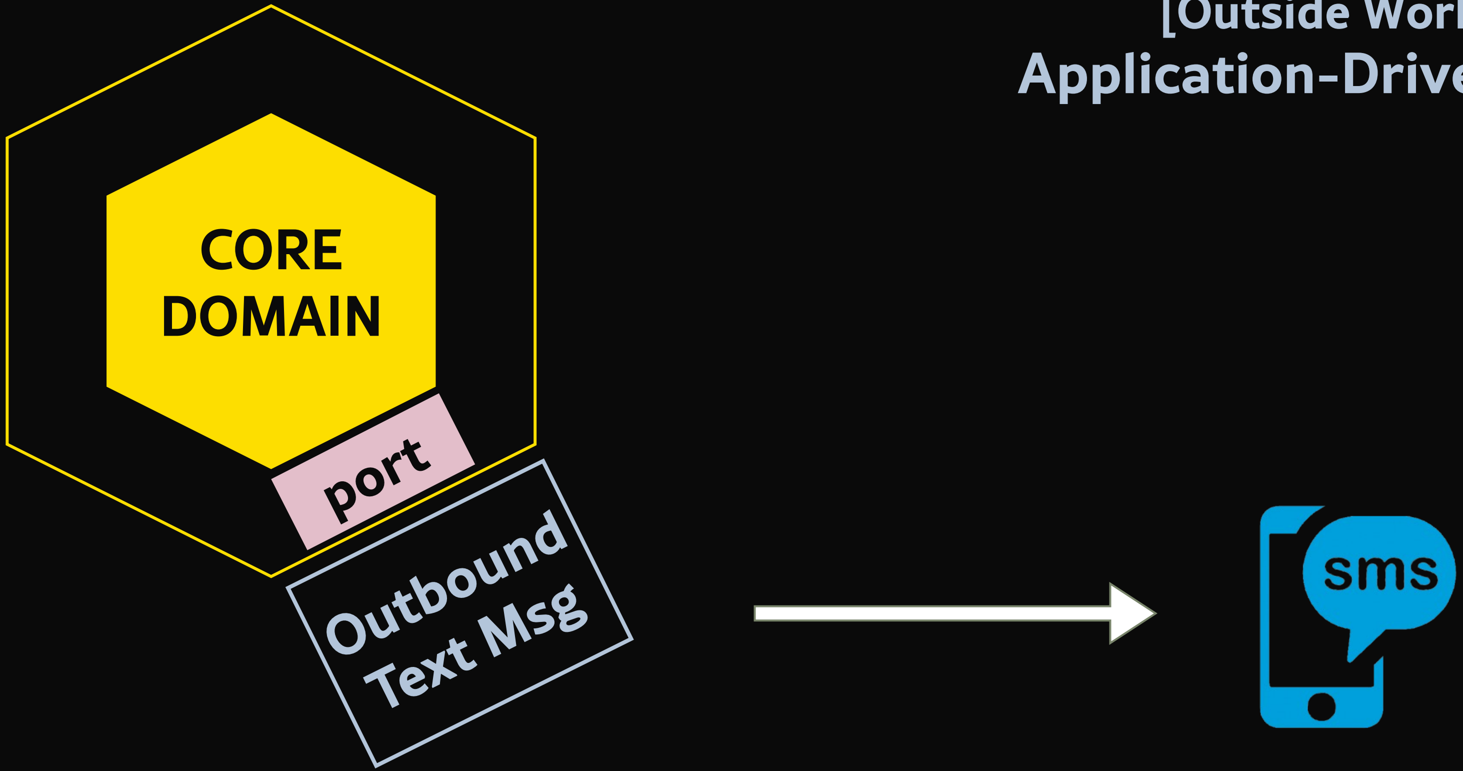
Ports

An *Interface* that

Provides a *Façade* for interacting with The Outside World

Implemented by the Outbound Adapter

[Outside World]
Application-Driven



Implementing Notifying Adapters

CREATE PORT
THE IDEAL INTERFACE

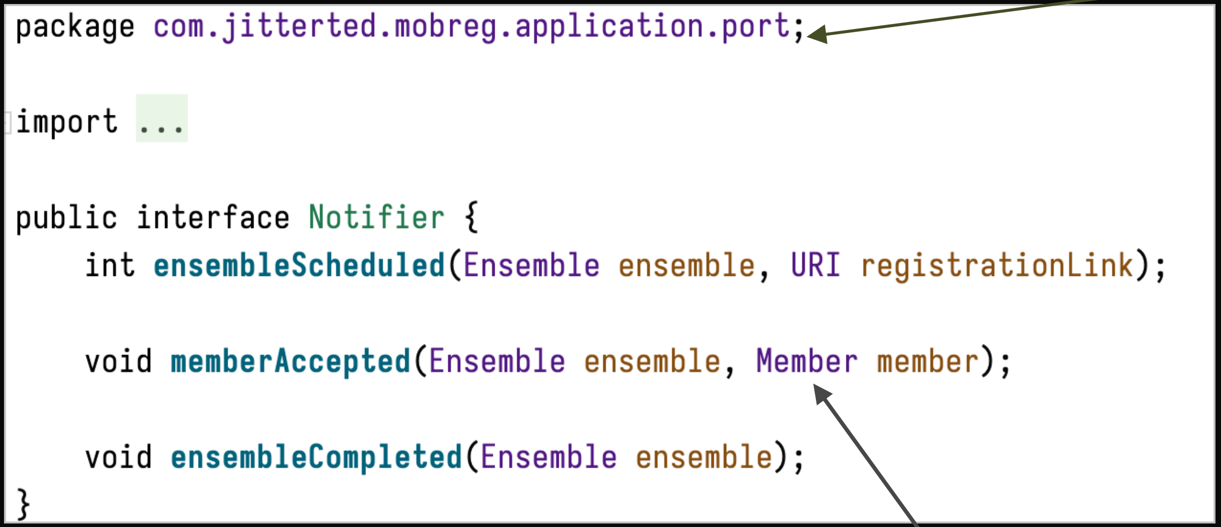
TEST DOMAIN CODE
USES SPY

REAL (CONCRETE) IMPLEMENTATION

Step 1: Define Port

Ideal Interface

Language of Domain



```
package com.jitterted.mobreg.application.port;

import ...

public interface Notifier {
    int ensembleScheduled(Ensemble ensemble, URI registrationLink);

    void memberAccepted(Ensemble ensemble, Member member);

    void ensembleCompleted(Ensemble ensemble);
}
```

in port package

Language of the Domain

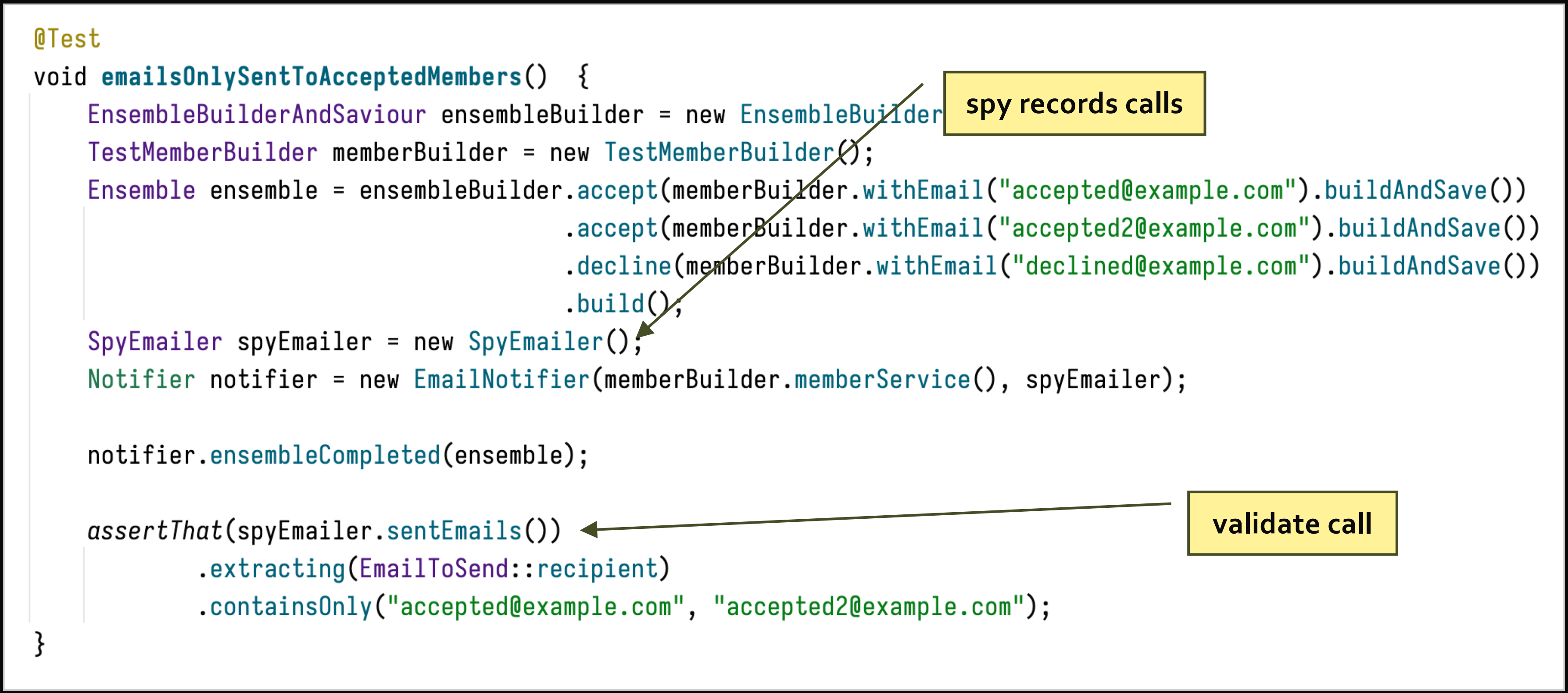
Step 2

Use Spy to Test

Spy

Records Method Calls

Behavior: Reports what happened



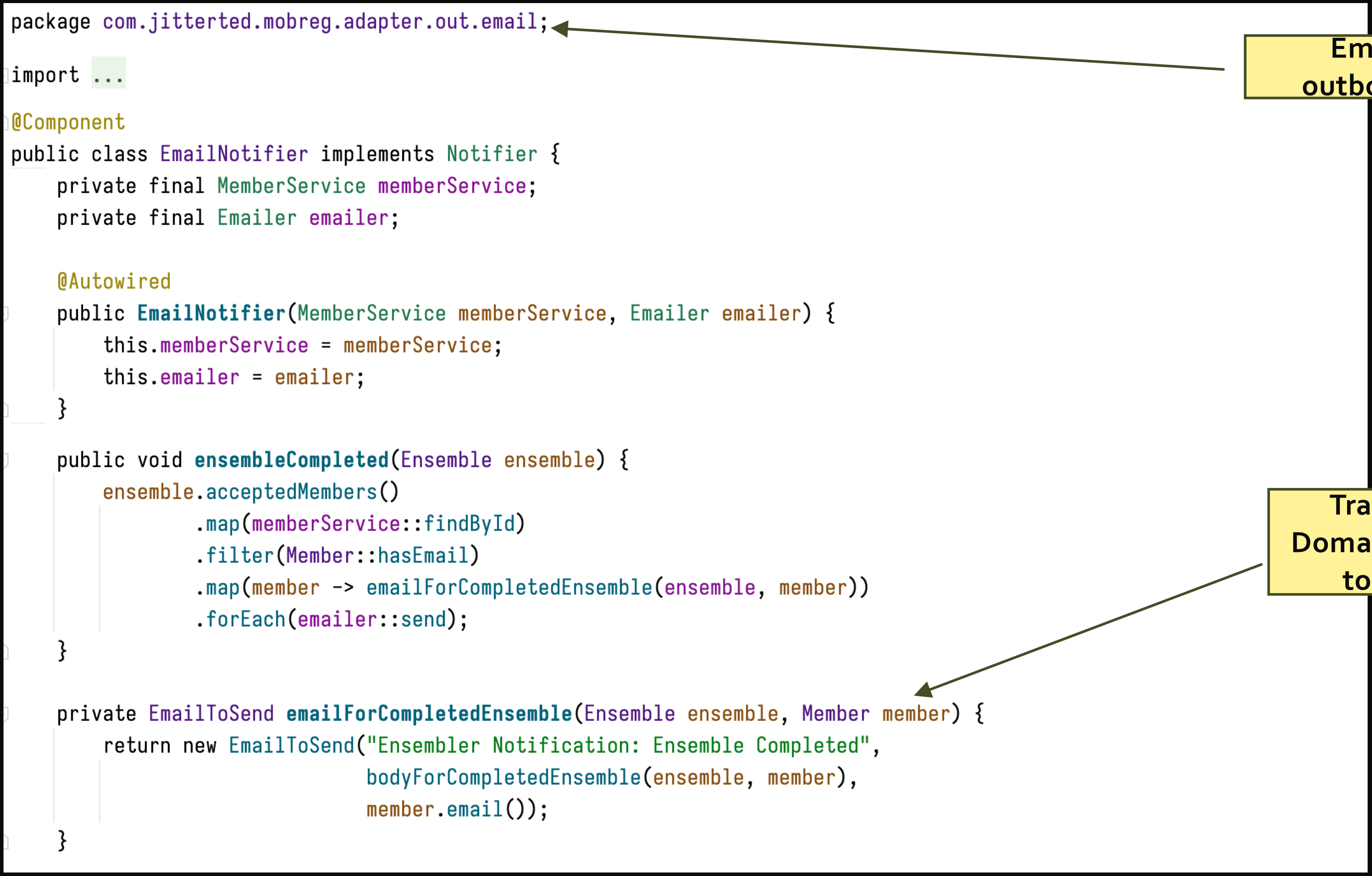
```java
@Test
void emailsOnlySentToAcceptedMembers()  {
    EnsembleBuilderAndSaviour ensembleBuilder = new EnsembleBuilder
    TestMemberBuilder memberBuilder = new TestMemberBuilder();
    Ensemble ensemble = ensembleBuilder.accept(memberBuilder.withEmail("accepted@example.com").buildAndSave())
                                       .accept(memberBuilder.withEmail("accepted2@example.com").buildAndSave())
                                       .decline(memberBuilder.withEmail("declined@example.com").buildAndSave())
                                       .build();
    SpyEmailer spyEmailer = new SpyEmailer();
    Notifier notifier = new EmailNotifier(memberBuilder.memberService(), spyEmailer);

    notifier.ensembleCompleted(ensemble);

    assertThat(spyEmailer.sentEmails())
            .extracting(EmailToSend::recipient)
            .containsOnly("accepted@example.com", "accepted2@example.com");
}
```

spy records calls

validate call

Step 3: Create Implementation

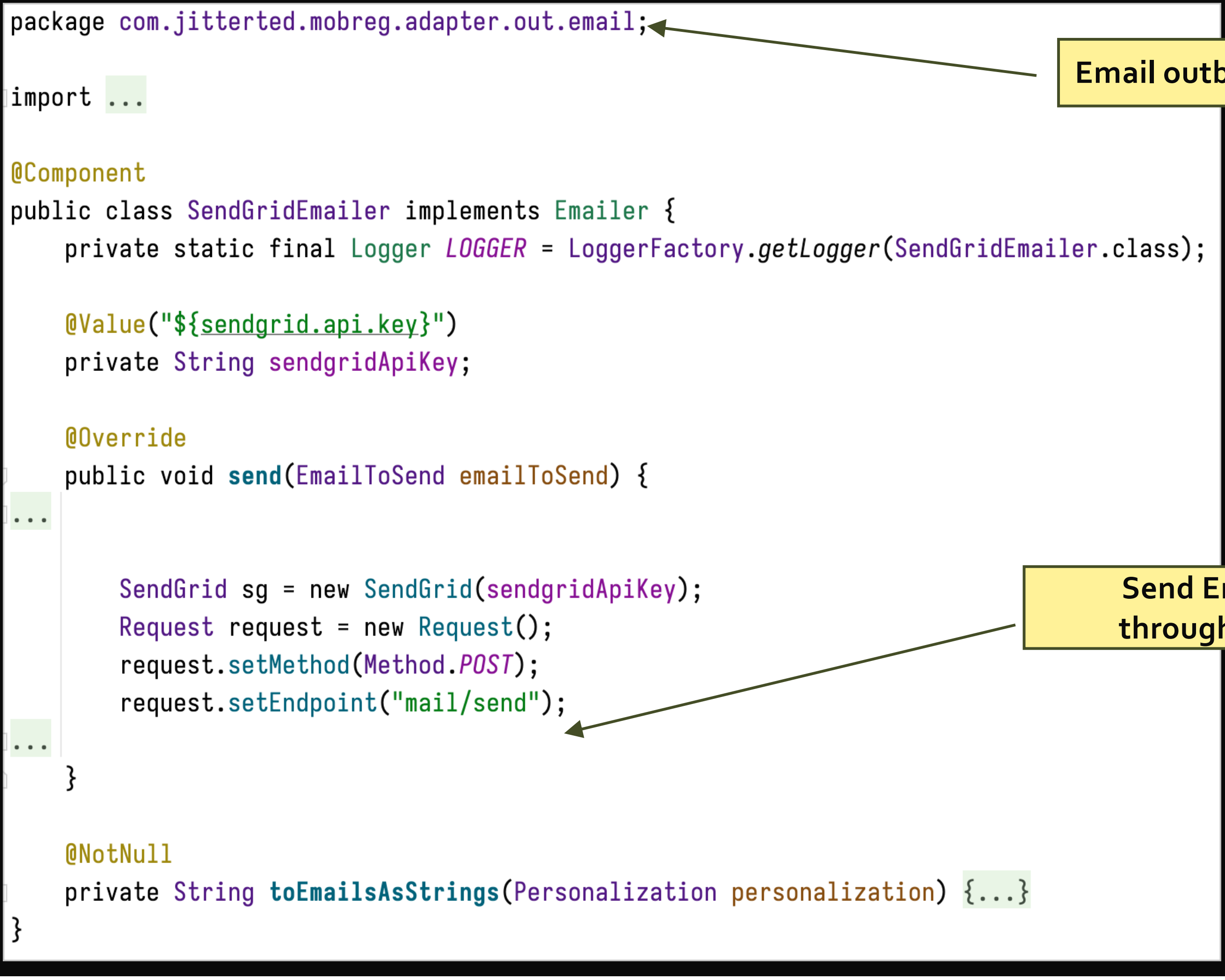
Concrete Adapter



```
package com.jitterted.mobreg.adapter.out.email;

import ...

@Component
public class EmailNotifier implements Notifier {
    private final MemberService memberService;
    private final Emailer emailer;

    @Autowired
    public EmailNotifier(MemberService memberService, Emailer emailer) {
        this.memberService = memberService;
        this.emailer = emailer;
    }

    public void ensembleCompleted(Ensemble ensemble) {
        ensemble.acceptedMembers()
                .map(memberService::findById)
                .filter(Member::hasEmail)
                .map(member -> emailForCompletedEnsemble(ensemble, member))
                .forEach(emailer::send);
    }

    private EmailToSend emailForCompletedEnsemble(Ensemble ensemble, Member member) {
        return new EmailToSend("Ensembler Notification: Ensemble Completed",
                               bodyForCompletedEnsemble(ensemble, member),
                               member.email());
    }
```

Email Notifier outbound adapter

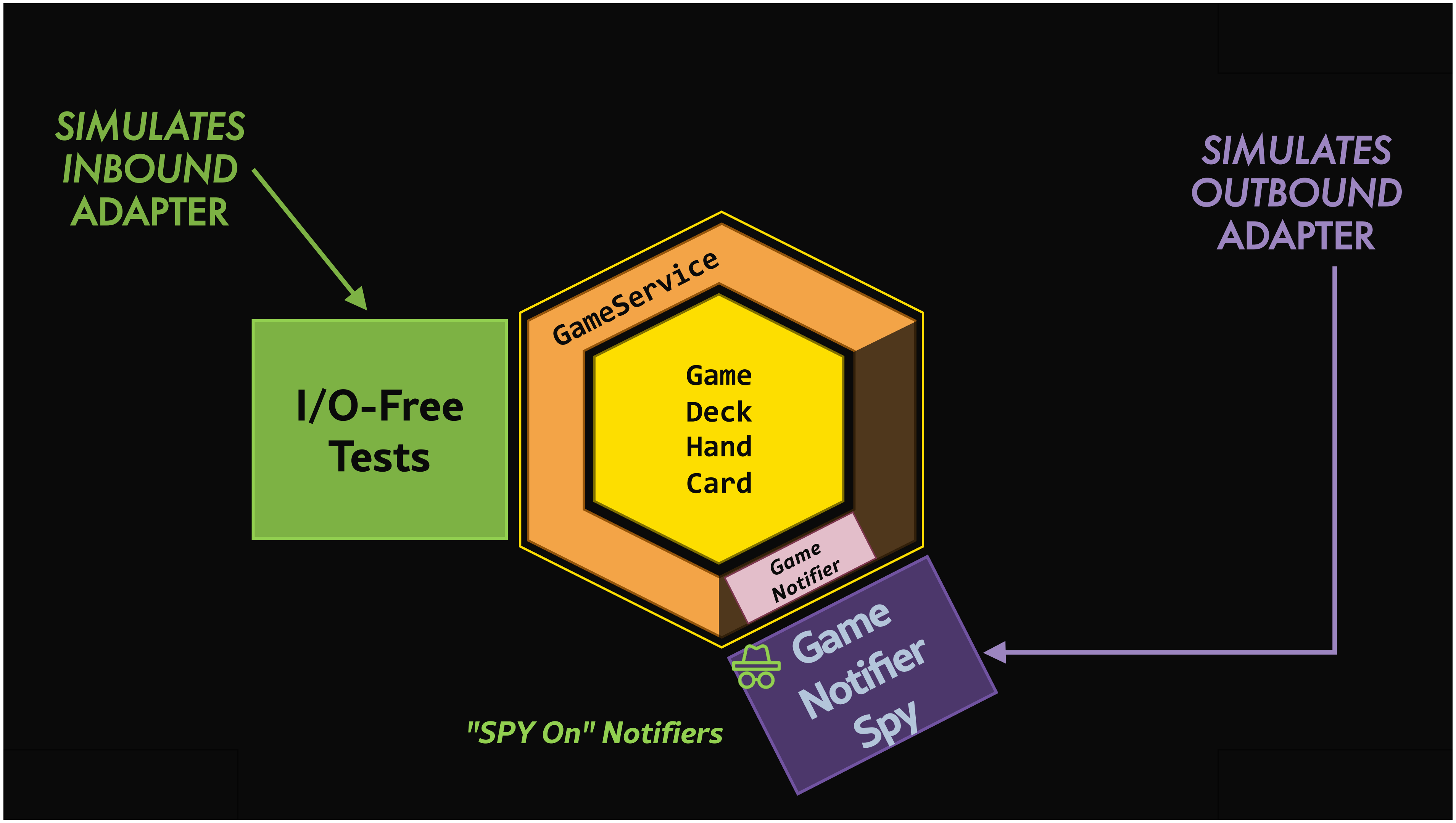
Transform Domain Objects to Email

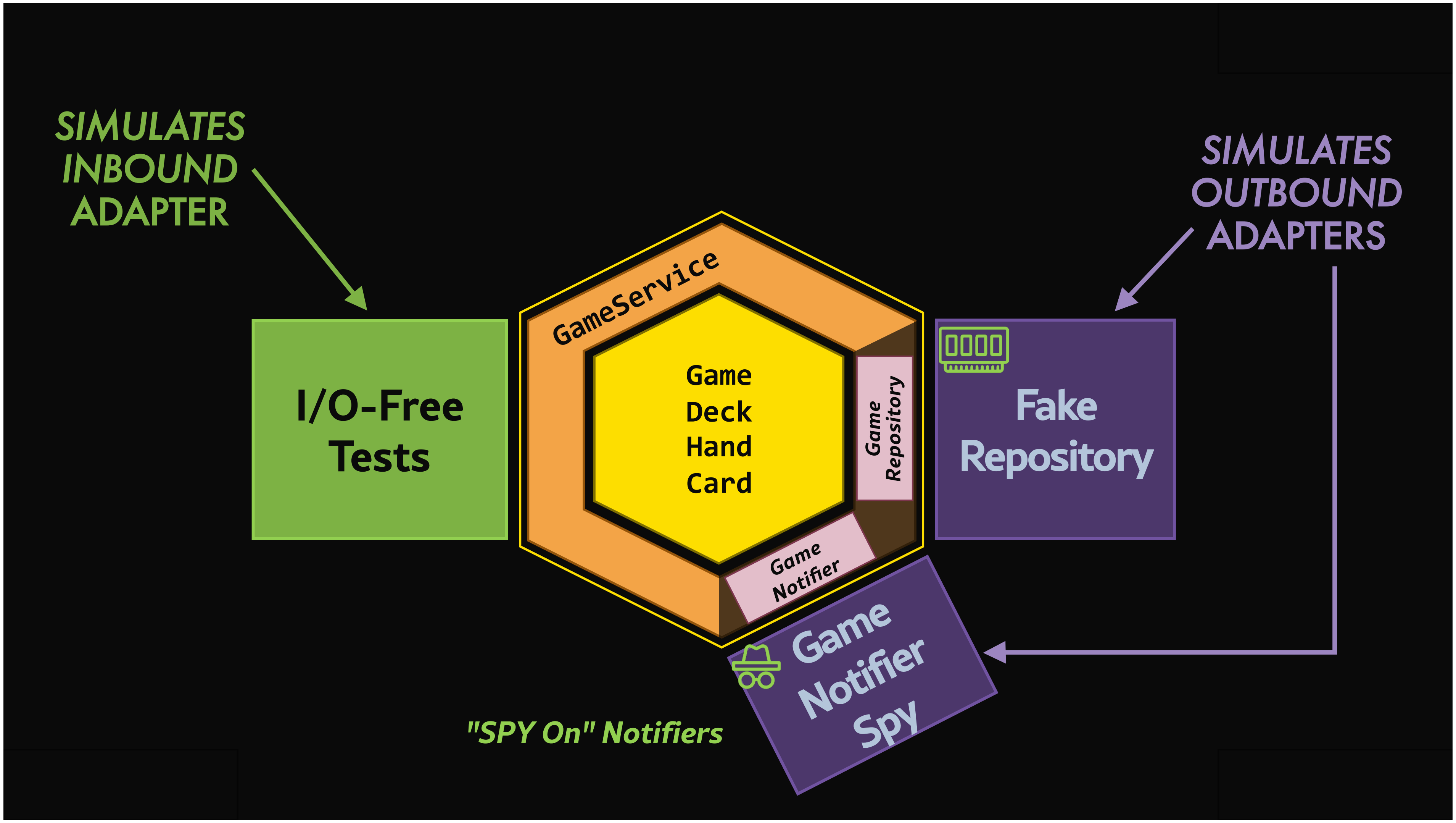


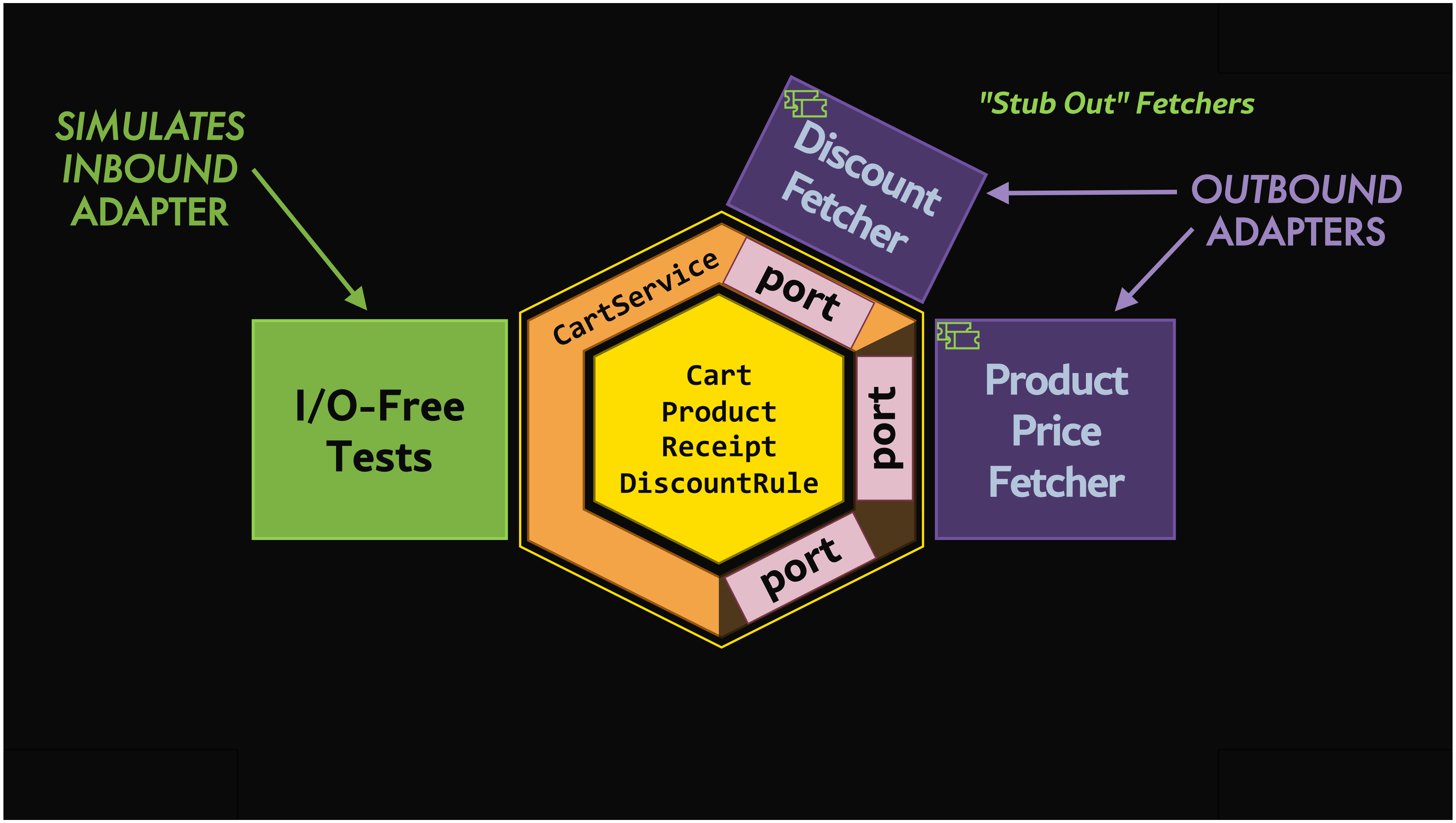
```
package com.jitterted.mobreg.adapter.out.email;

import ...

@Component
public class SendGridEmailer implements Emailer {
    private static final Logger LOGGER = LoggerFactory.getLogger(SendGridEmailer.class);

    @Value("${sendgrid.api.key}")
    private String sendgridApiKey;

    @Override
    public void send(EmailToSend emailToSend) {
...

        SendGrid sg = new SendGrid(sendgridApiKey);
        Request request = new Request();
        request.setMethod(Method.POST);
        request.setEndpoint("mail/send");
...
    }

    @NotNull
    private String toEmailsAsStrings(Personalization personalization) {...}
}
```

Email outbound adapter

Send Email through I/O


SIMULATES INBOUND ADAPTER
I/O-Free Tests
GameService
Game
Deck
Hand
Card
Game Notifier
SIMULATES OUTBOUND ADAPTER
Game Notifier Spy
"SPY On" Notifiers


SIMULATES INBOUND ADAPTER
I/O-Free Tests
GameService
Game
Deck
Hand
Card
Game Repository
Game Notifier
SIMULATES OUTBOUND ADAPTERS
Fake Repository
Game Notifier Spy
"SPY On" Notifiers


SIMULATES INBOUND ADAPTER
I/O-Free Tests
CartService
Cart
Product
Receipt
DiscountRule
port
port
port
Discount Fetcher
"Stub Out" Fetchers
OUTBOUND ADAPTERS
Product Price Fetcher

Repository

Used for Storing and Retrieving Aggregates

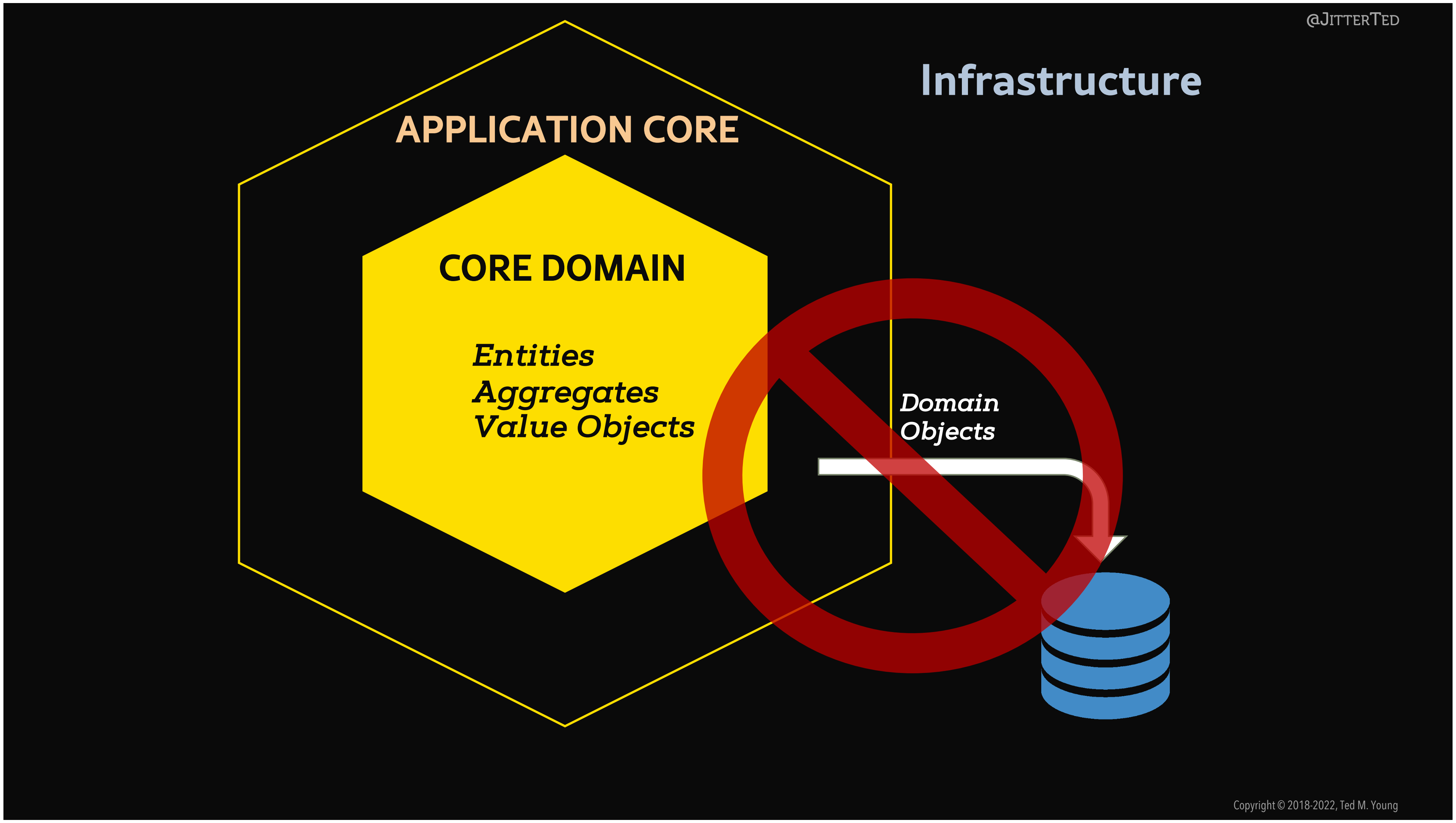
Rule

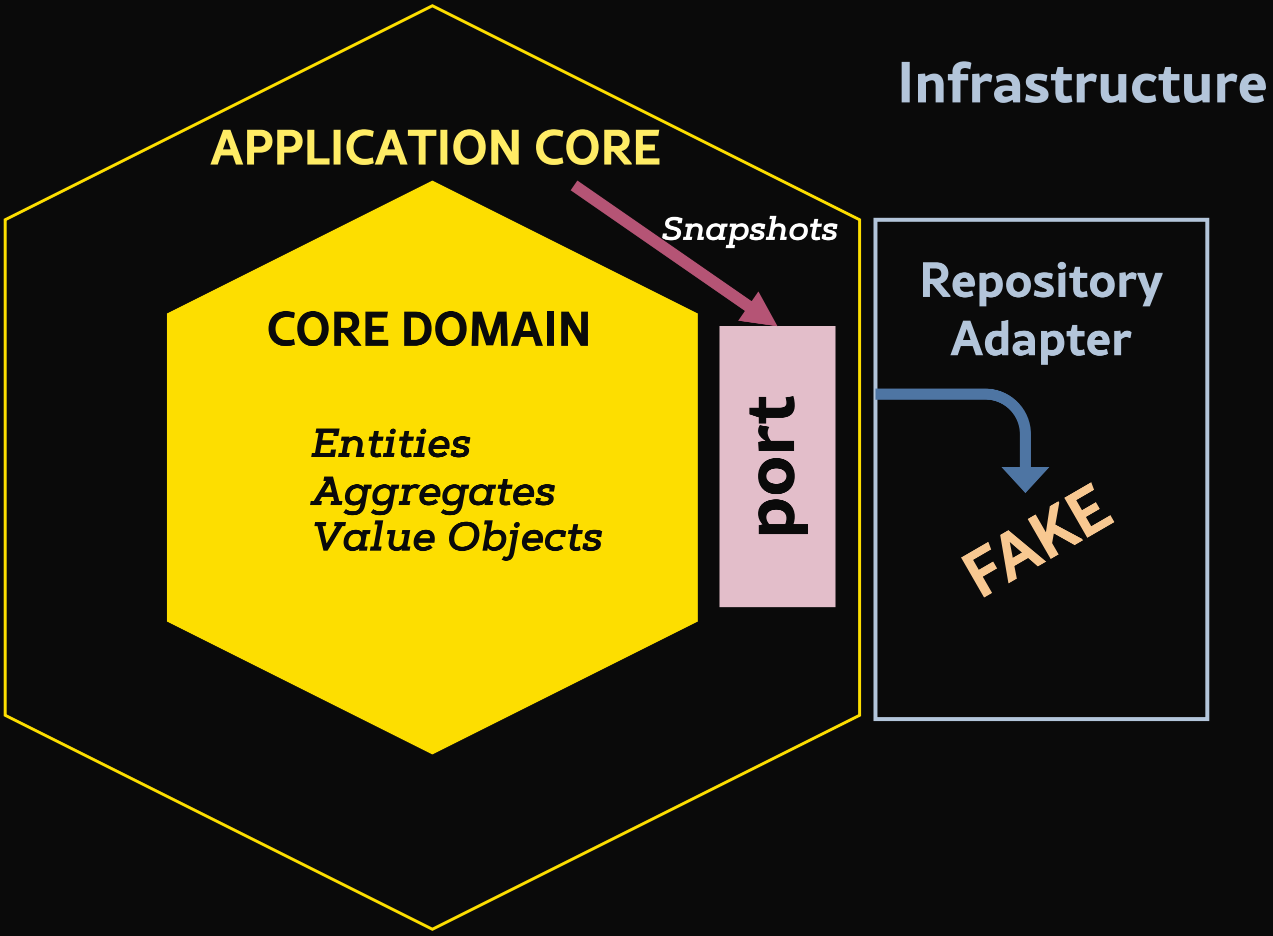
Persistence Ignorance

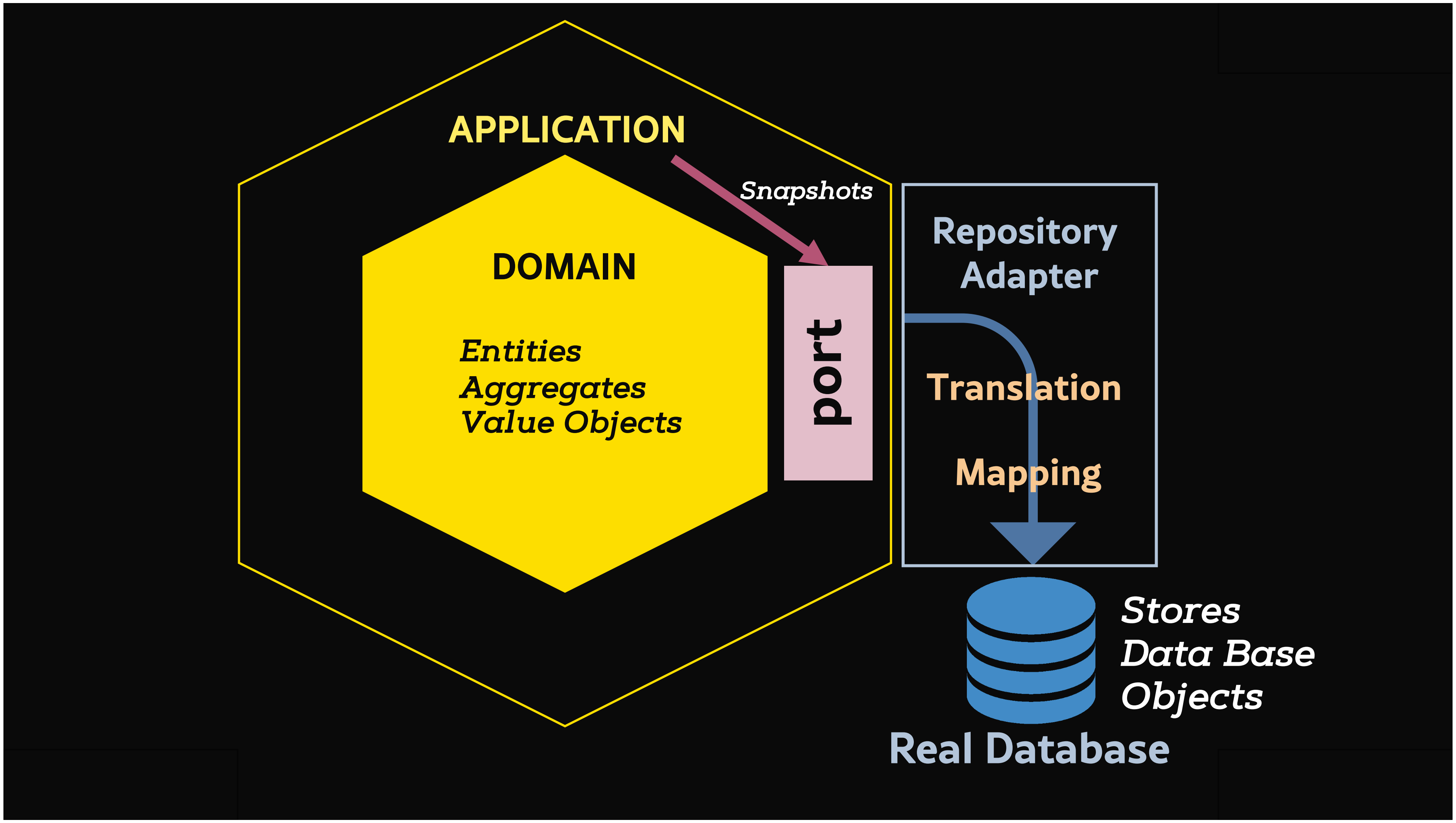
Core Domain Objects

Ensemble
Member
ConferenceDetails
EnsembleStatus
MemberStatus
Rsvp

EnsembleRepository
MemberRepository


Infrastructure
APPLICATION CORE
CORE DOMAIN
Entities
Aggregates
Value Objects
Domain
Objects


Infrastructure
APPLICATION CORE
Snapshots
CORE DOMAIN
Entities
Aggregates
Value Objects
port
Repository Adapter
FAKE


APPLICATION
DOMAIN
Entities
Aggregates
Value Objects
Snapshots
port
Repository
Adapter
Translation
Mapping
Stores
Data Base
Objects
Real Database

Don't Use Hexagonal

Basic CRUD

Simple or non-existent domain

Transformation/Integration-only

Hexagonal Architecture

Focus on Core Domain

Domain Objects Never Leave

Adapters per Trigger Type

Ports for Ideal Interface to Outside

Dependencies point inwards

Package Based on Architecture

What Questions Do You Have?

Ted M. Young
Java Trainer, Coach
& Live Coder

✓ TESTABLE ✗
HEXAGONAL ARCHITECTURE
ORGANIZED

Twitter: @JitterTed
Email: ted@tedmyoung.com
Live Coding: https://JitterTed.Stream

Web: https://TED.dev
YouTube: https://JitterTed.TV

Visit r2ha.com

Ted M. Young
Java Trainer, Coach
& Live Coder

✓ TESTABLE ✗

HEXAGONAL ARCHITECTURE

ORGANIZED

Twitter: @JitterTed
Email: ted@tedmyoung.com
Live Coding: https://JitterTed.Stream
Web: https://TED.dev
YouTube: https://JitterTed.TV