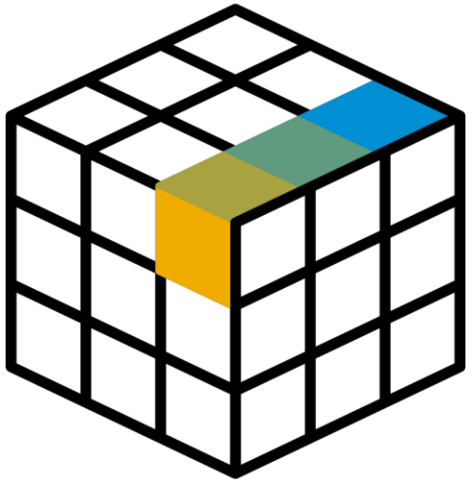
Week 1: Introducing SAP Cloud Application Programming Model

Unit 1: Introduction to SAP Cloud Application Programming Model

Motivation

- Customers need a stable environment for business application development
- Reduce high entry barrier for less tech-savvy developers
- Customers ask for guidance and best practices
- Developers want to focus on business logic

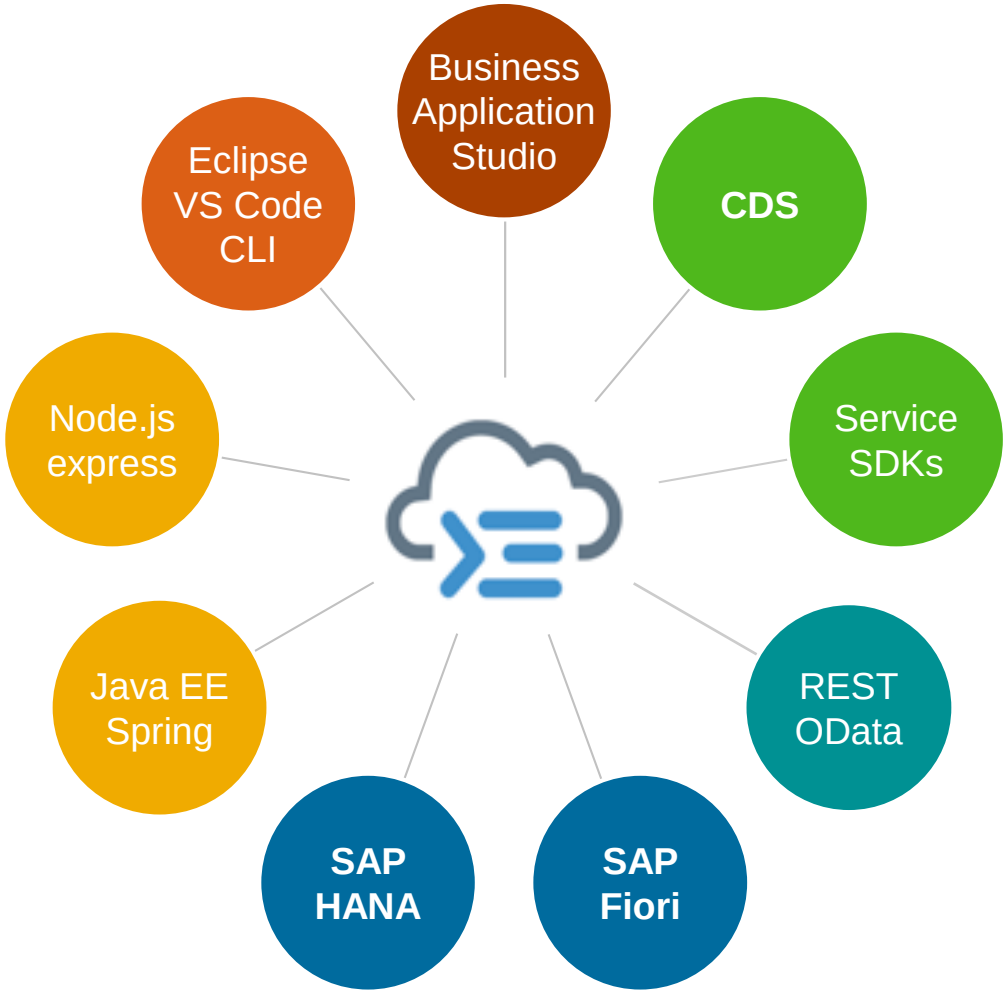


Introduction

*The SAP Cloud Application Programming Model is an **opinionated**, yet **open** framework of tools, languages, and libraries to efficiently build enterprise-grade services and applications.*

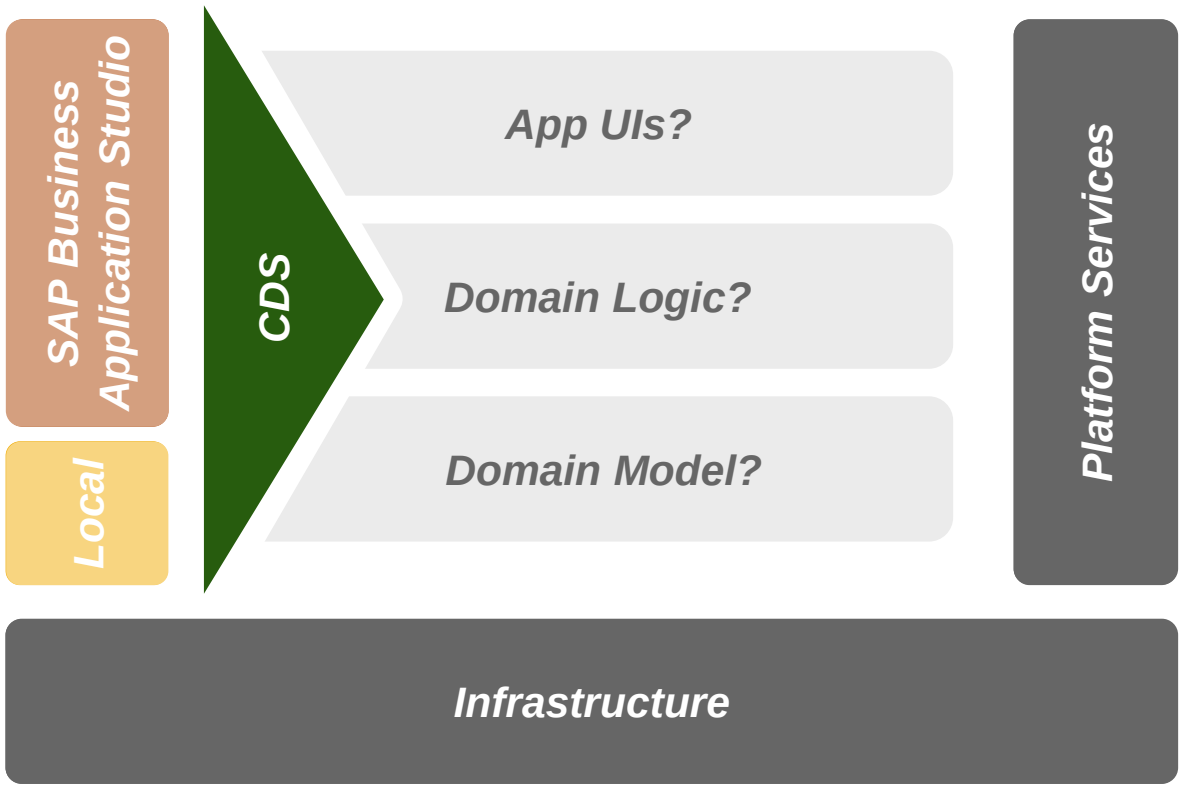
*It guides developers along a 'golden path' of proven **best practices**, while minimizing boilerplate so they can **focus on their domain** problems at hand.*

*The framework features a mix of broadly adopted **open-source** and **SAP** tools and technologies.*



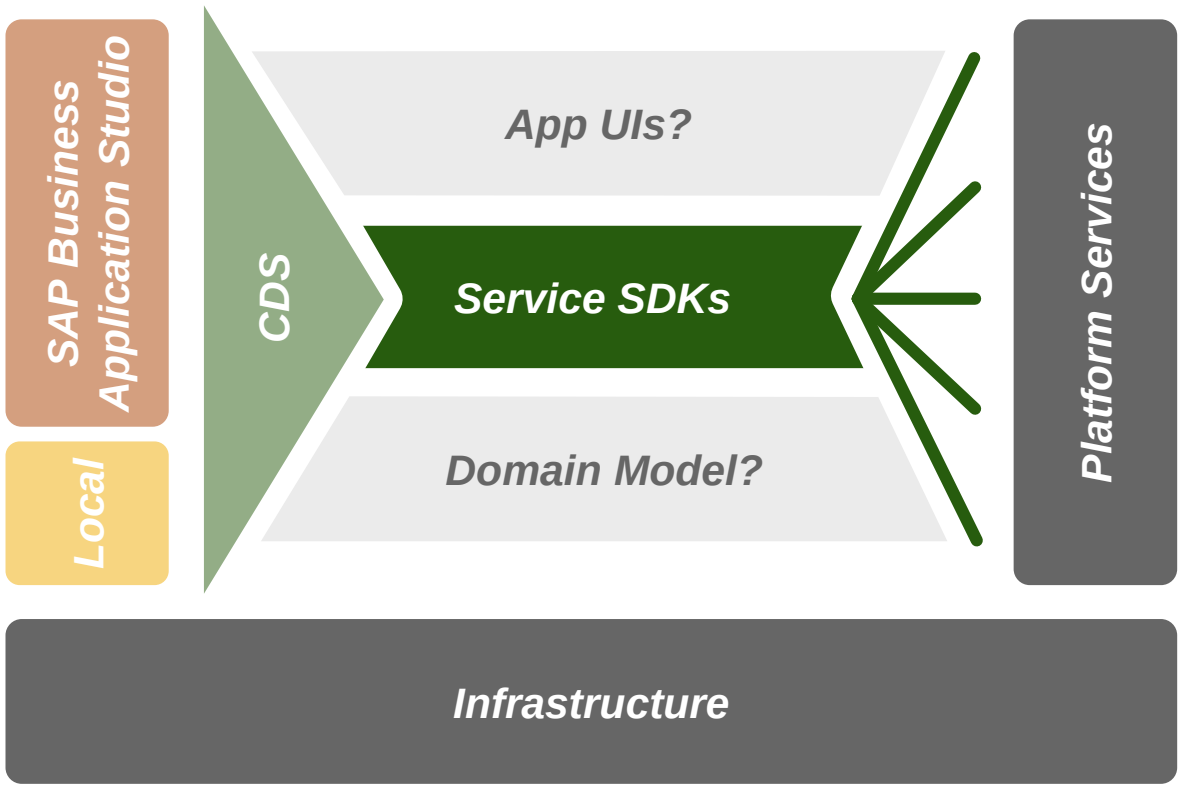
Ingredients

- **Core Data Services (CDS)** is a data modeling infrastructure
- You can use CDS to define data models and service definitions



Ingredients

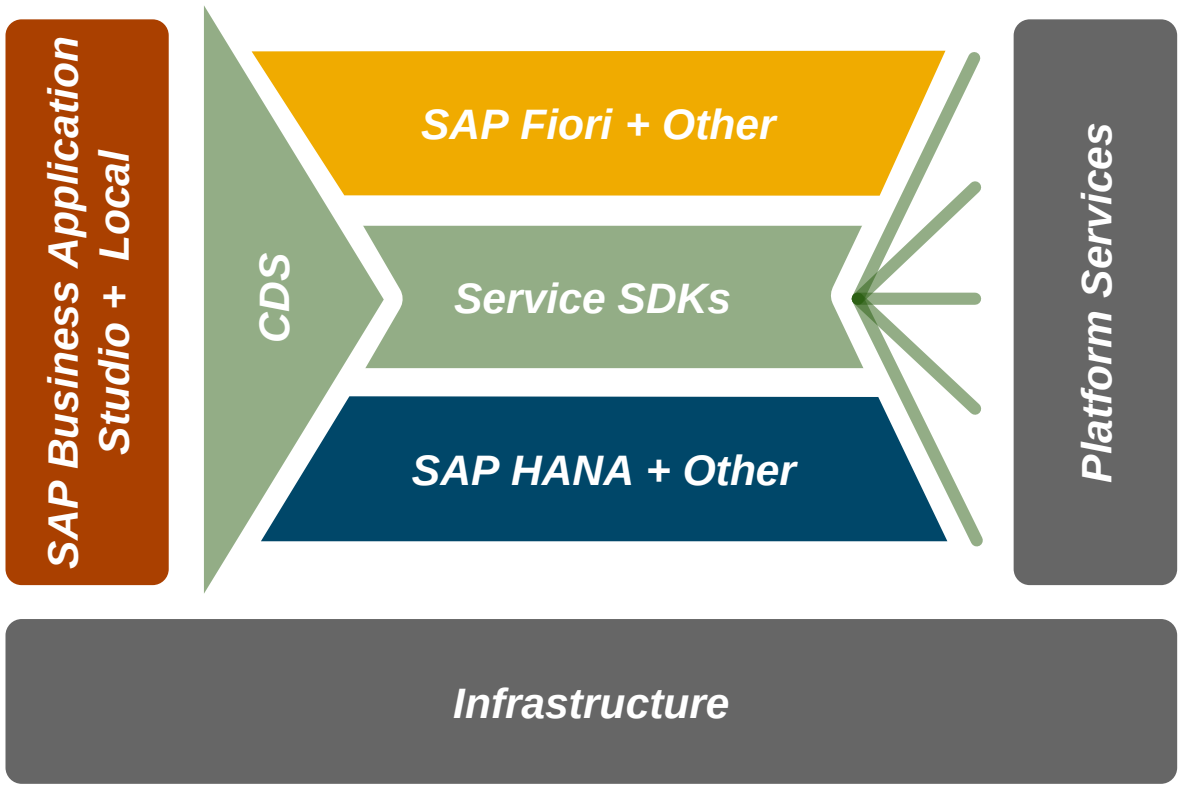
- Choose between **Java** and **Node.js**
- Provide and consume services
- Easy access to libraries provided by SAP
- High-level integration to low-level platform services (abstraction)



Ingredients

Get *1st-class support* for…

- **SAP Business Application Studio** with tailored tools
- **SAP Fiori Elements**
- **SAP HANA**



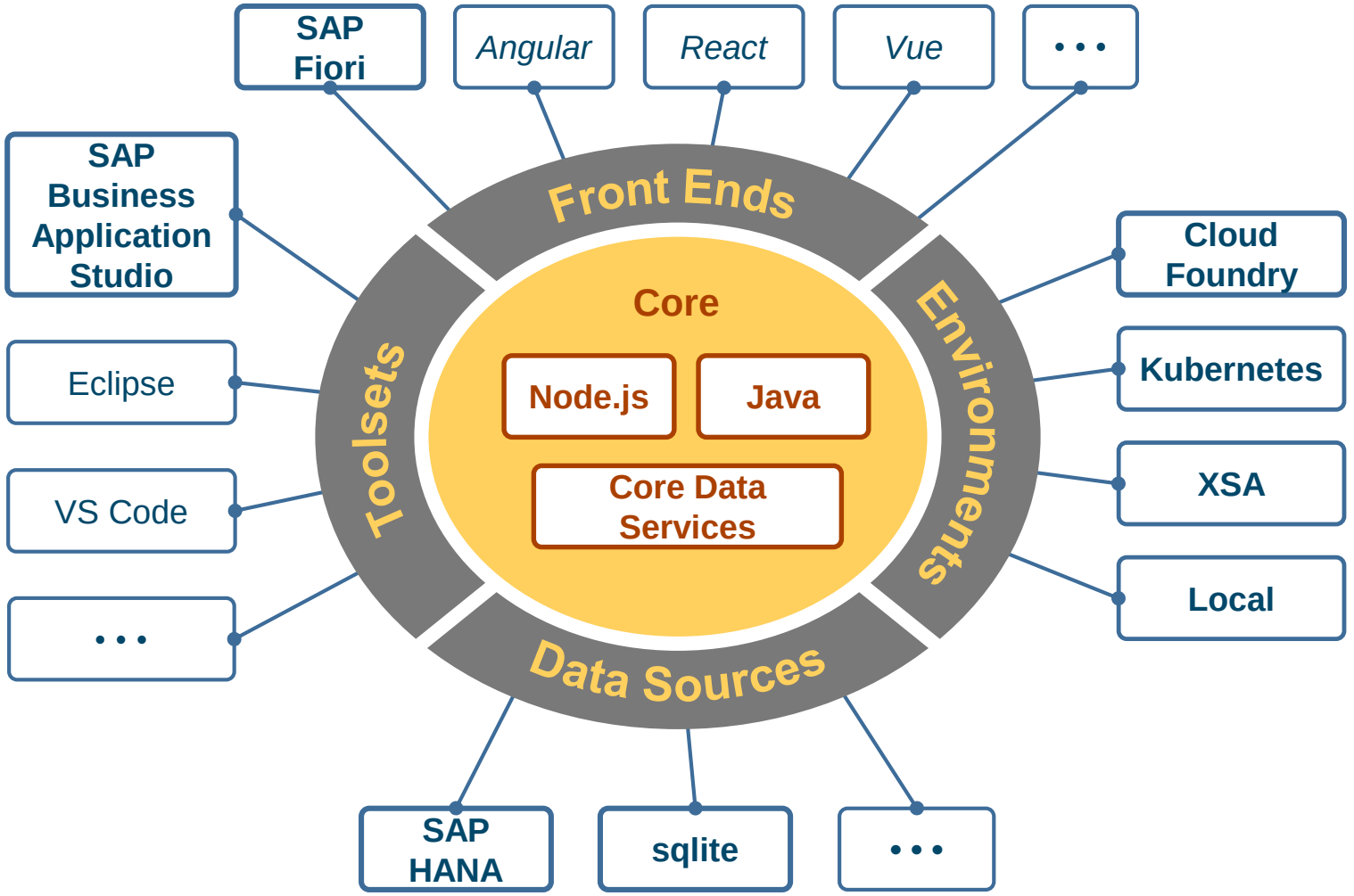
Design principles

Open

- You choose technologies
- Select your architecture pattern
- Use CAP or parts of CAP

Opinionated

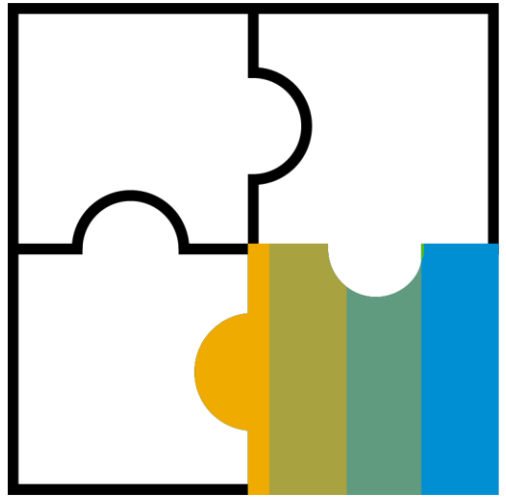
- Gives guidance
- Support for recurring tasks



Design principles

Zero Lock-In

- You choose UI and database technology
- SAP Fiori Elements and SAP HANA out-of-the-box
- Designed to integrate with alternatives



Design principles

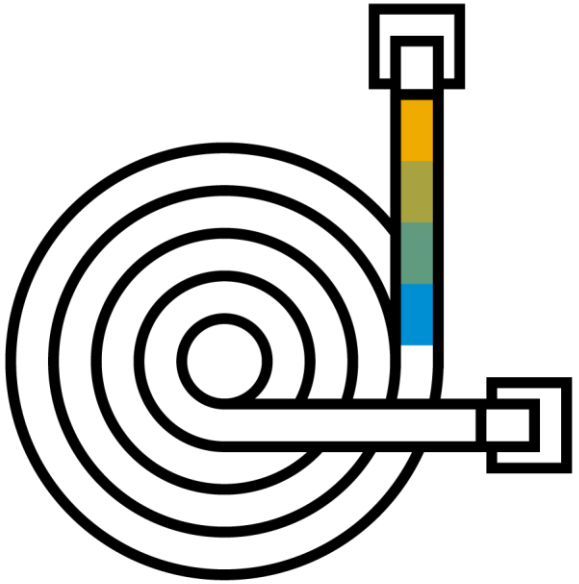
Non-Intrusive

- Your architecture, your methodology
- CAP is free of assumptions
- Avoids boilerplate using convention over configuration

Design principles

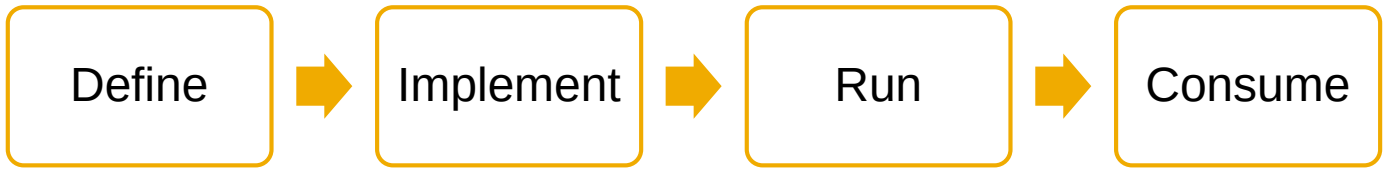
Platform-Agnostic

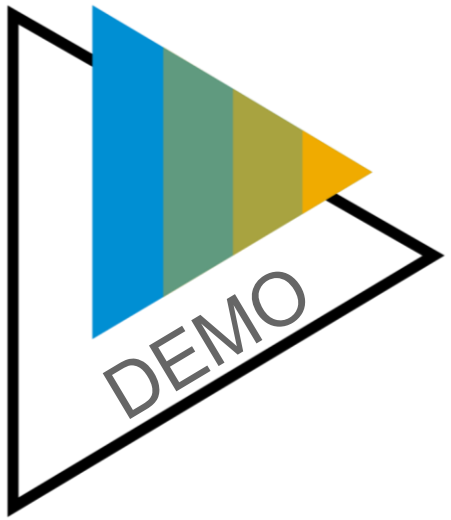
- Use higher-level APIs
- Avoids hard-wiring
- Enables you to keep pace with cloud technologies



Demo

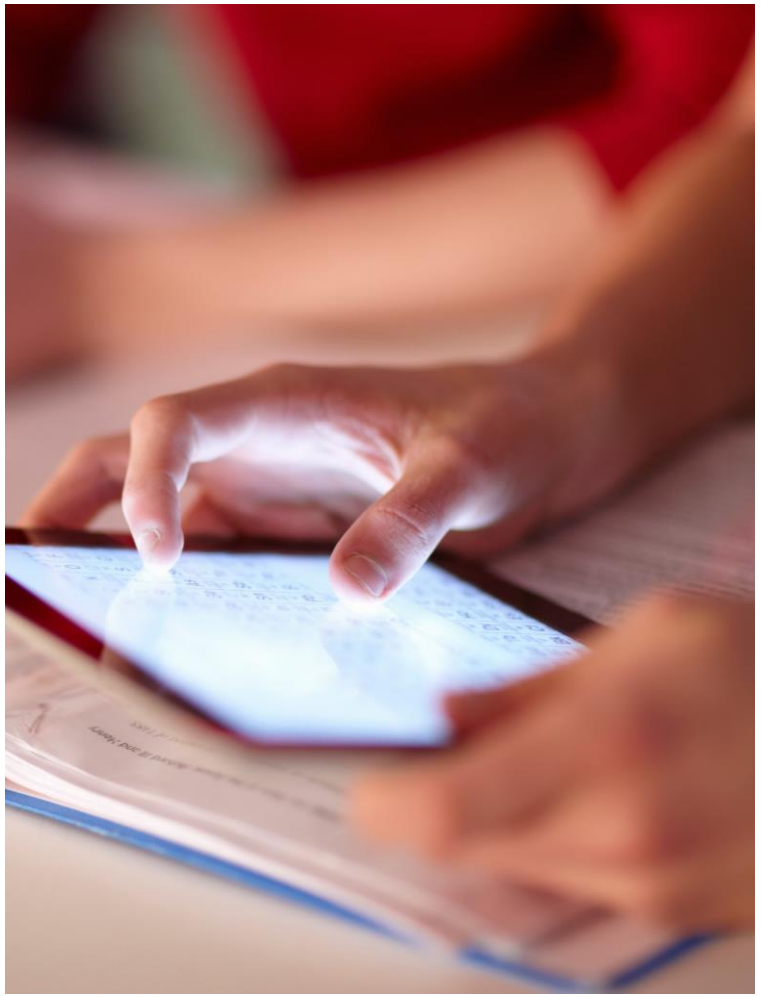
Create a “**Hello World**” application





What you've learned in this unit

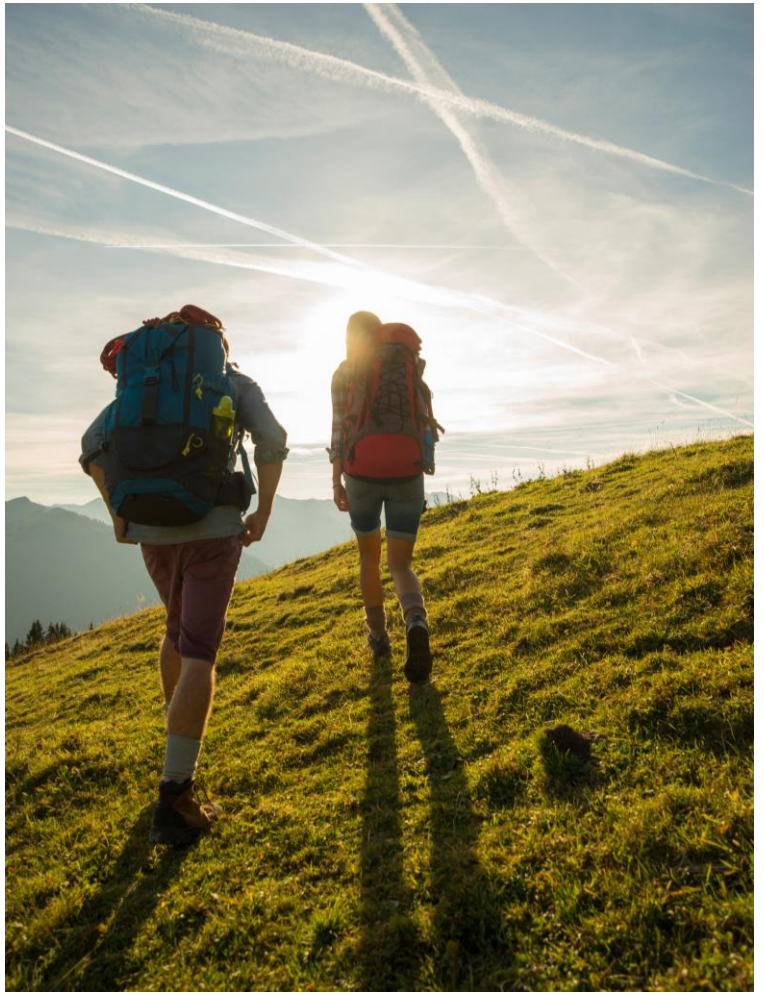
- How you can focus on your business domain by using SAP Cloud Application Programming Model
- What benefits you have while developing with SAP Cloud Application Programming Model
- Which toolset and capabilities SAP Cloud Application Programming Model offers for development



Further reading

Additional Material

- Official documentation
- Blogs Cloud Application Programming Model
- Starter Scenario
- TechEd Replay
- Hello World Tutorial



Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

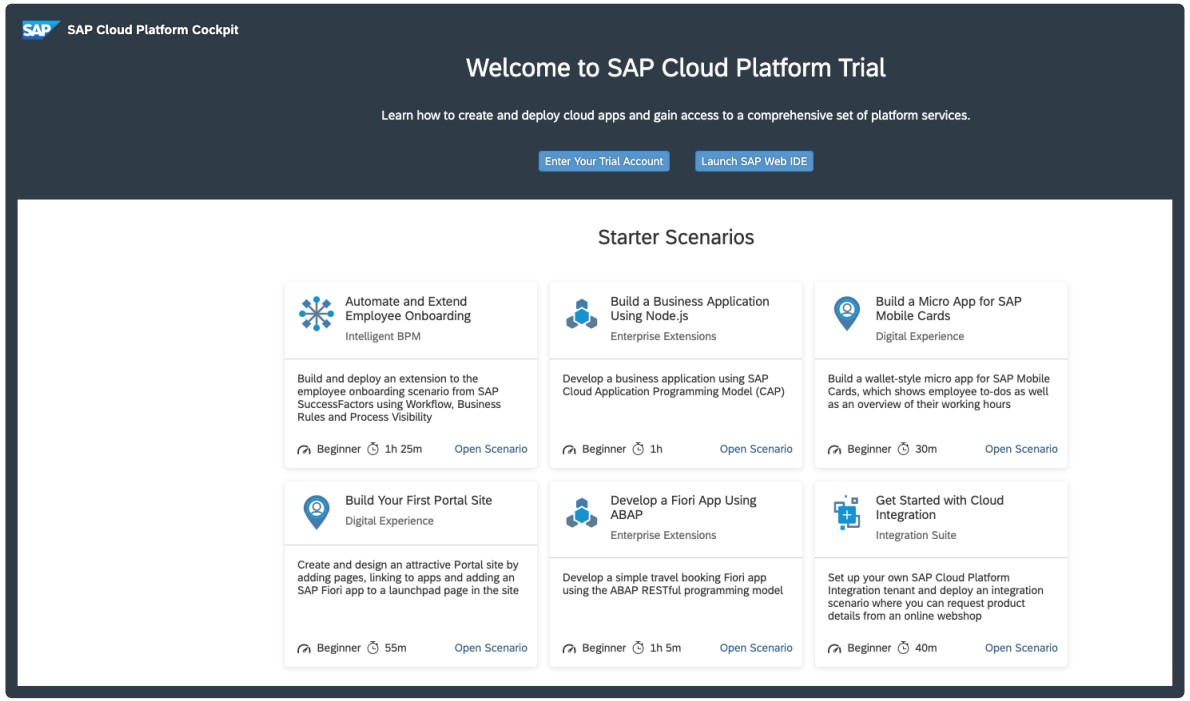
THE BEST RUN SAP®

Week 1: Introducing SAP Cloud Application Programming Model

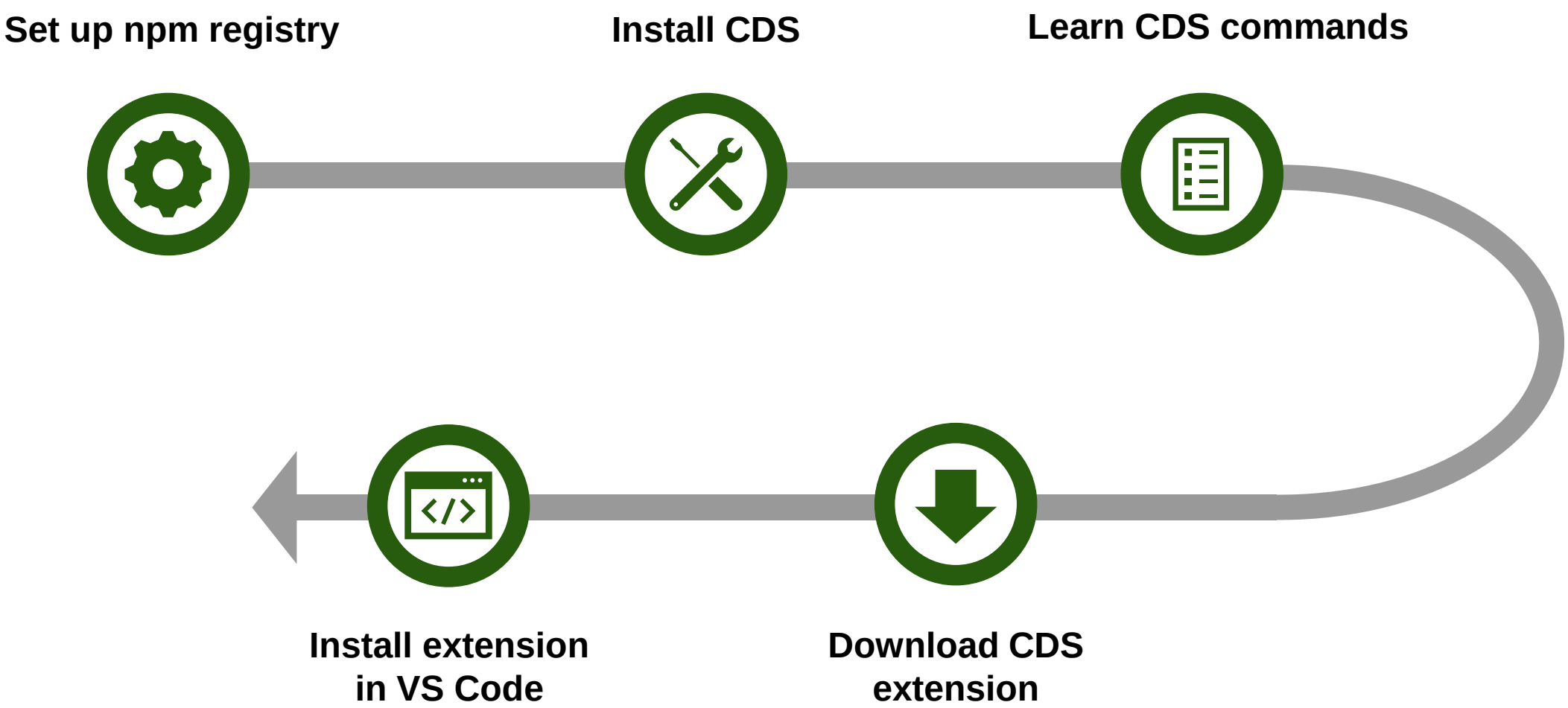
Unit 2: Tools and Setup for Core Data Services (CDS)

Getting an SAP Cloud Platform trial account

- SAP Cloud Platform account to try out features
- Free of charge
- Expires after 30 days, but can be extended to a limit of 90 days
- Use production and beta services
- **NO** service-level agreement with regard to the availability of the platform

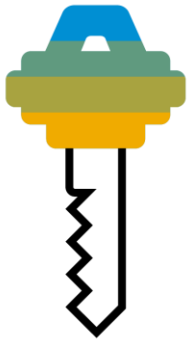


Using local development tools



SAP Business Application Studio

A modular driven environment, tailored for efficient development of business applications for the Intelligent Enterprise



Turn-key solutions

Pre-configured development environments tailored for SAP specific scenarios

Ultimate Productivity

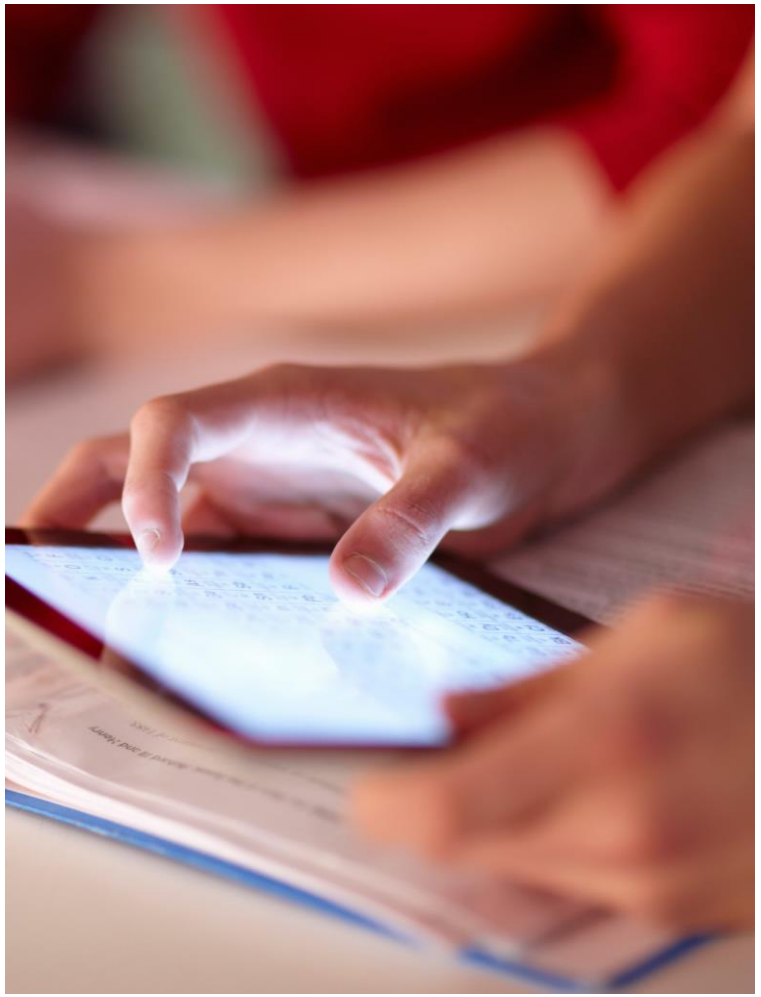
End-to-end development platform, with a desktop-like experience in the cloud

Integrate SAP solutions

Integrating SAP services and SAP solutions via built-in connectivity

What you've learned in this unit

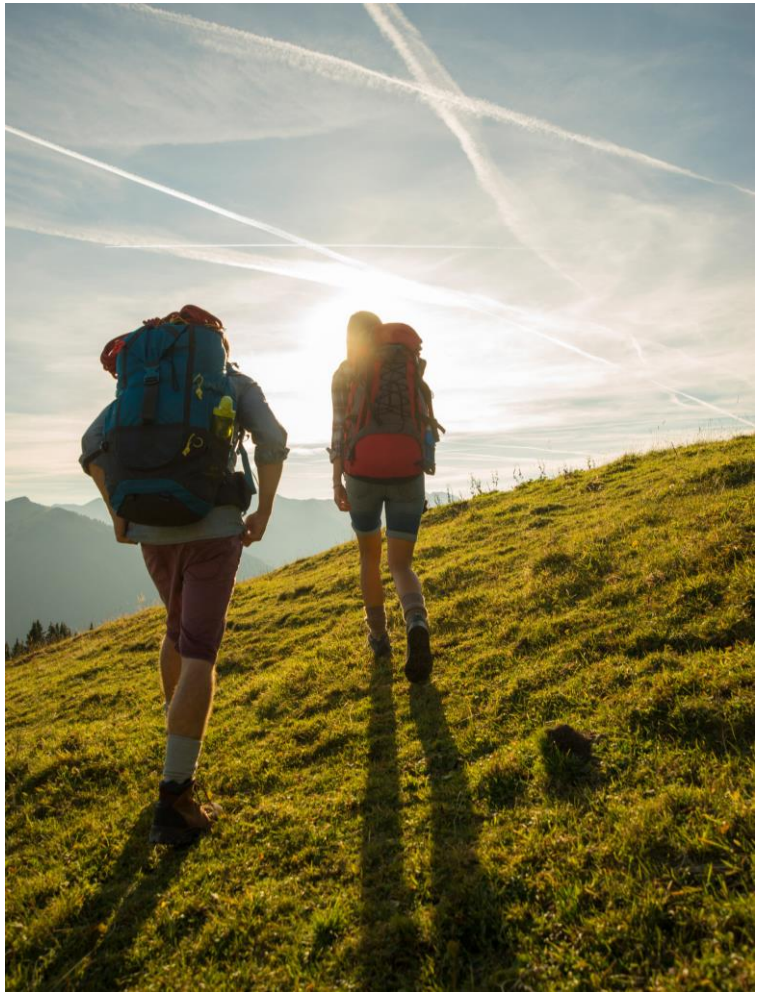
- How to create an SAP Cloud Platform trial account
- The limitations of SAP Cloud Platform trial accounts
- How to set up your local development environment
- How to develop applications with SAP Business Application Studio



Further reading

Additional Material

- Documentation:
  Trial Accounts in the Cloud Foundry Environment
- Website: Getting a trial account
- Tutorial: Setup local development tools
- Blog: SAP Business Application Studio
- Documentation: SAP Business Application Studio



Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

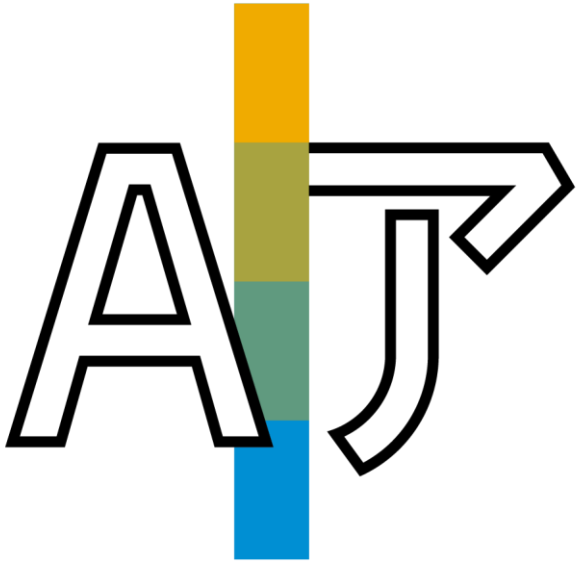
THE BEST RUN SAP®

Week 1: Introducing SAP Cloud Application Programming Model

Unit 3: Getting Started with Core Data Services (CDS)

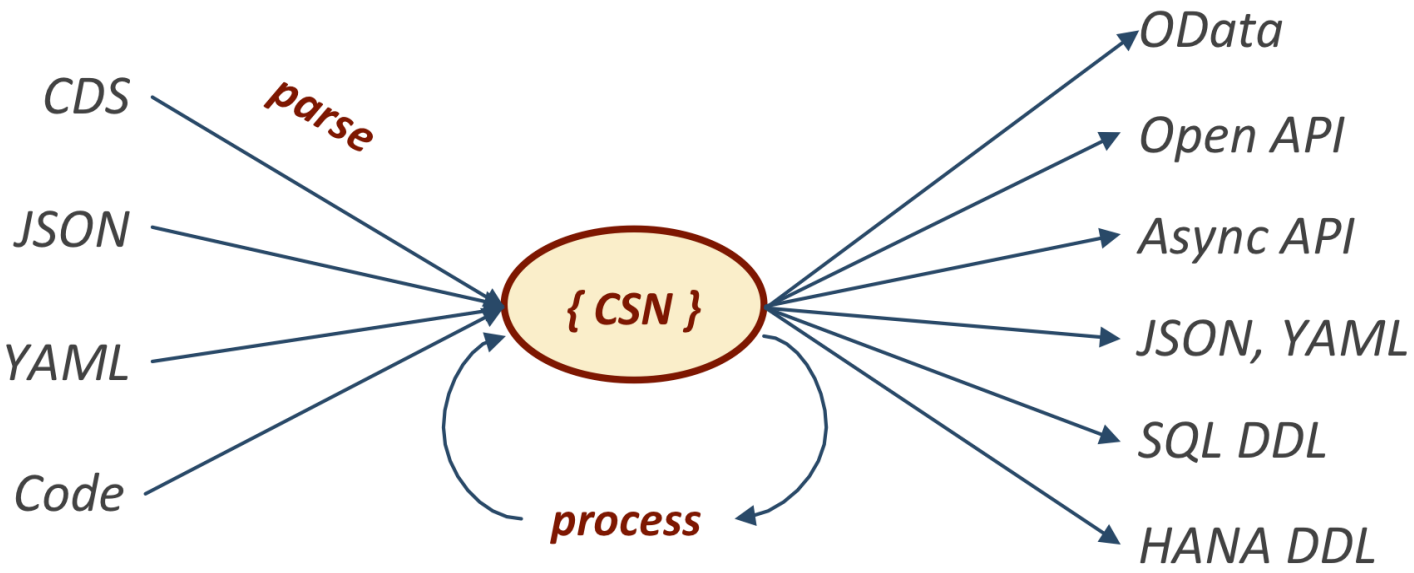
What is CDS?

- CDS stands for Core Data Services
- It is a set of domain-specific languages
- The backbone of the SAP Cloud Application Programming Model



What is CDS?

- Declaratively capture data models, service definitions, queries, and expressions
- JavaScript objects complying with the Core Schema Notation (CSN)
- Models can be easily created and interpreted, which fosters extensions by 3rd party contributions



Definition Language (CDL)

- Entities
- Views
- Associations
- Aspects
- Types
- Service definitions
- … and more

```
entity Books: managed @(title:'Bücher') {
  key ID : UUID;
  title  : localized String;
  author : Association to Authors;
}

entity Authors {
  key ID : UUID;
  name   : String;
  books  : Composition of many Books;
}
```

Definition Language (CDL)

- Entities
- Views
- Associations
- Aspects
- Types
- Service definitions
- … and more

```
entity Books as projection on db.Books;

entity Authors as projection on db.Authors;

entity Orders as SELECT * from db.Orders;
```

Definition Language (CDL)

- Entities
- Views
- Associations
- Aspects
- Types
- Service definitions
- … and more

```
entity Books: managed @(title:'Bücher') {
  key ID : UUID;
  title  : localized String;
  author : Association to Authors;
}

entity Authors {
  key ID : UUID;
  name   : String;
  books  : Composition of many Books;
}
```

Definition Language (CDL)

- Entities
- Views
- Associations
- Aspects
- Types
- Service definitions
- … and more

```
aspect managed {
  modifiedAt : DateTime @cds.on.update : $now;
  createdAt  : DateTime @cds.on.insert : $now;
  createdBy  : User     @cds.on.insert : $user;
  modifiedBy : User     @cds.on.update : $user;
}

entity Books: managed @(title:'Bücher') {
  key ID : UUID;
  title  : localized String;
  author : Association to Authors;
}
```

Definition Language (CDL)

- Entities
- Views
- Associations
- Aspects
- Types
- Service definitions
- … and more

```
type User : String(111);

type Amount {
  value : Decimal(10,3);
  currency : Currency;
}

type Currency : Association to Currencies;

entity Order {
  buyer : User;
  price : Amount;
}
```

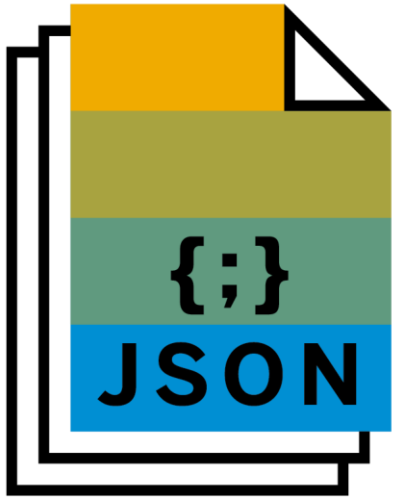
Definition Language (CDL)

- Entities
- Views
- Associations
- Aspects
- Types
- Service definitions
- … and more

```
service CatalogService {
  entity Books as projection on db.Books;
  entity Authors as projection on db.Authors;

  entity Orders as projection on db.Orders {
    *,
    book.title,
    book.author.name as author
  } where createdBy = $user;

  action cancel(order:UUID);
}
```

Core Schema Notation (CSN)

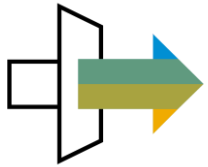
- A compact JSON representations of data and service models
- Similar to JSON Schema but goes beyond with the ability to capture full-blown entity-relationship models
- Perfect source to generate target models, such as OData/EDM interfaces, as well as persistence models for SQL or NoSQL databases.



Core Schema Notation (CSN)

CDL entity into CSN example:

```
entity Books {

  key ID : UUID;

  title  : String;

  author : Association to Authors;

}
```

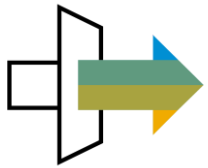


```
"Books": {
 "kind": "entity",
 "elements": {
  "ID":{ "key": true, "type": "cds.UUI" },
  "title": { "type": "cds.String" },
  "author": {
   "type": "cds.Association",
   "target": "Authors",
   "keys": [ { "ref": [ "id" ] }]
  }
 }
}
```

Core Schema Notation (CSN)

CDL service with an action into CSN example:

```
service Orders {

 action cancelOrder(orderID: Integer)

  returns {

   ack: Boolean

         enum { succeeded; failed; };

   msg: String;

 };

}
```

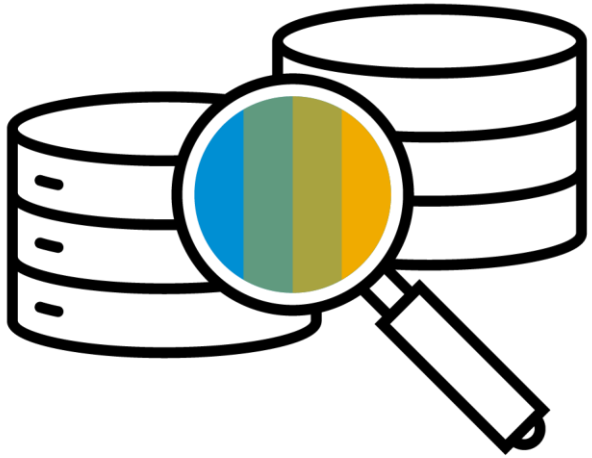


```
"Orders": {
      "kind": "service"
},
"Orders.cancelOrder": {
 "kind": "action",
 "params": {
  "orderID": { "type": "cds.Integer" }
 },
 "returns": {
  "elements": {
   "ack": {
    "type": "cds.Boolean",
    "enum": { "succeeded": {},
              "failed": {}
            }
    },
   "msg": { "type": "cds.String" }
  }
 }
}
```

Query Language (CQL)

CDS QL is based on standard SQL with some enhancements:

- Postfix projections
- Path expressions
- CDL-style casts
- Excluding clause
- Query-local mixins



Query Language (CQL)

- Postfix projections
- Path expressions
- CDL-style casts
- Excluding clause
- Query-local mixins

```
SELECT from Books { *, author.name as author};
```

Query Language (CQL)

- Postfix projections
- Path expressions
- CDL-style casts
- Excluding clause
- Query-local mixins

```
SELECT from Authors[name='Emily Brontë'].books;

SELECT from Books:authors.towns;

SELECT title, author.name from Books;

SELECT from Books where author.name='Emily
Brontë';
```

Query Language (CQL)

- Postfix projections
- Path expressions
- CDL-style casts
- Excluding clause
- Query-local mixins

```
SELECT from Books { stock+1 as bar : Decimal };
```

Query Language (CQL)

- Postfix projections
- Path expressions
- CDL-style casts
- Excluding clause
- Query-local mixins

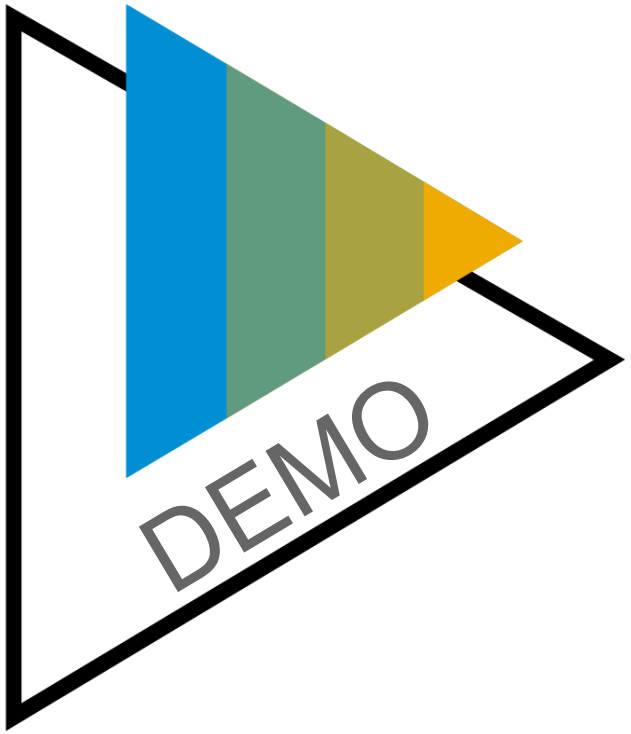
```
SELECT from Books { * } excluding { author };
```

Query Language (CQL)

- Postfix projections
- Path expressions
- CDL-style casts
- Excluding clause
- Query-local mixins

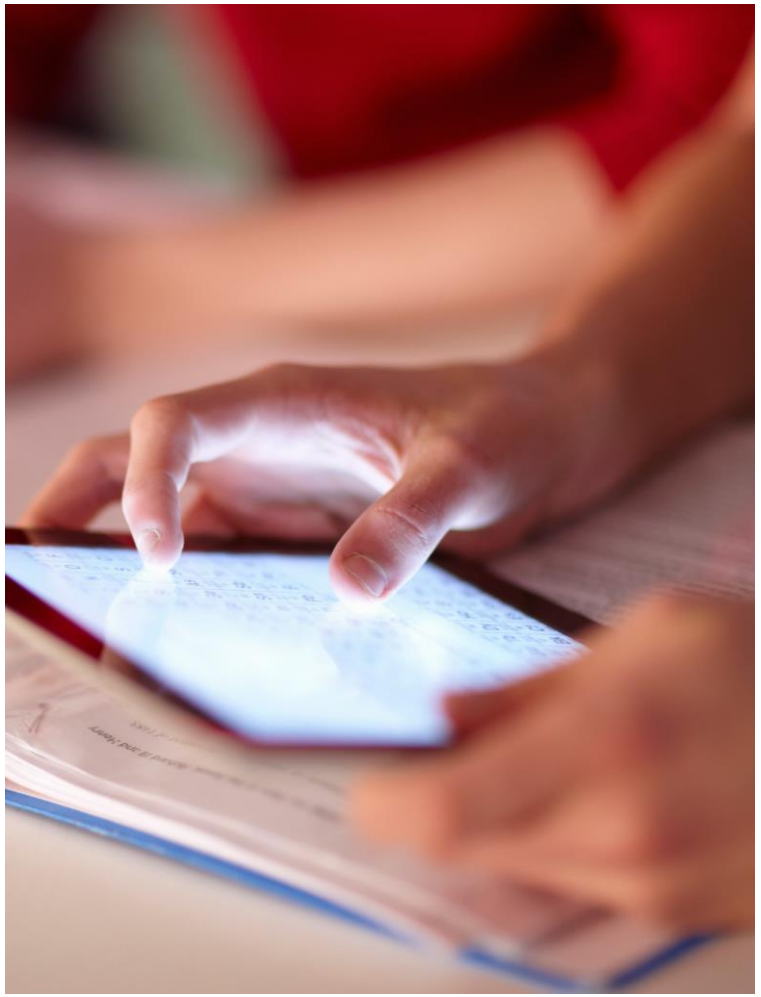
```
SELECT from Books mixin {

 localized : Association to LocalizedBooks on
localized.ID = ID;

} into {

 ID, localized.title

};
```

Demo



What you've learned in this unit

- What CDS is
- Definition Language (CDL)
- Core Schema Notation (CSN)
- Query Language (CQL)



Further reading

Additional Material

- Official documentation
- Definition Language (CDL)
- Core Schema Notation (CSN)
- Query Language (CQL)
- Domain-specific languages guide

Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

THE BEST RUN SAP®

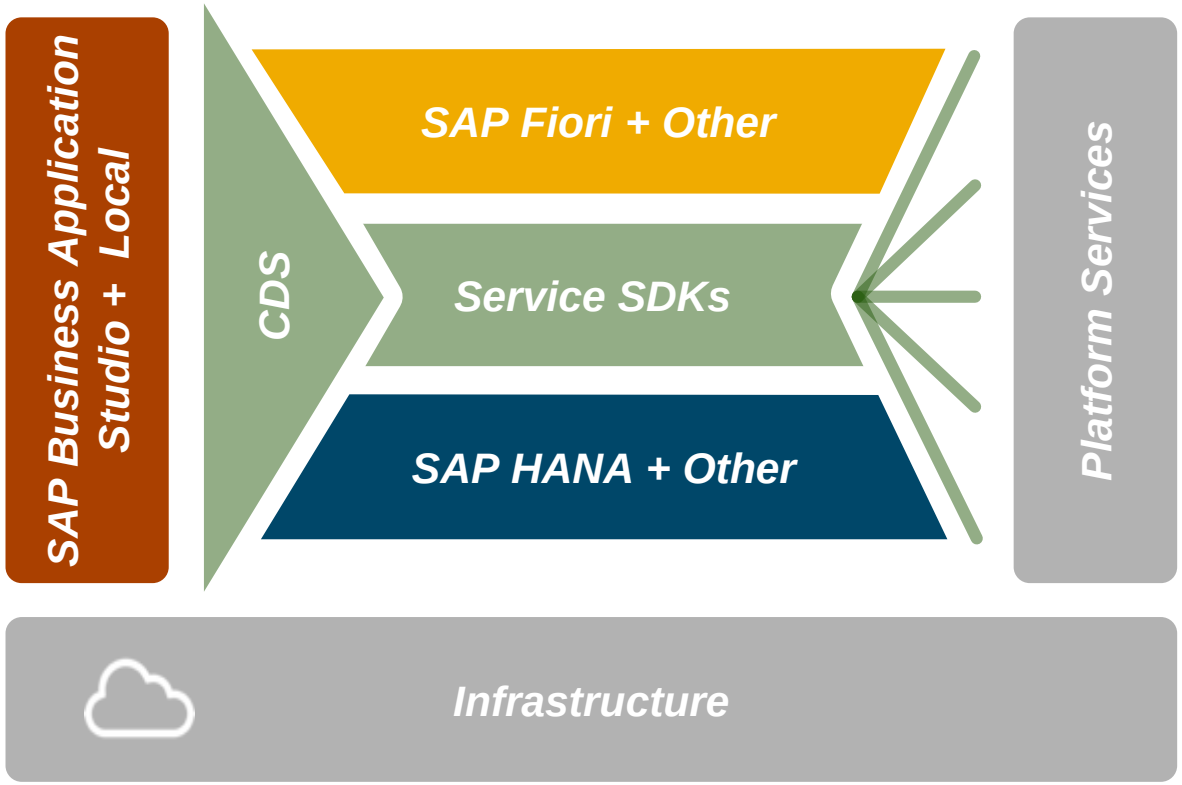
Week 1: Introducing SAP Cloud Application Programming Model

Unit 4: Your First App Using Node.js with SAP Cloud Application Programming Model

Node.js for business service development

Node.js

- Asynchronous event-driven JavaScript runtime
- Built on top of Chrome's V8 engine
- One language for front and back-end development

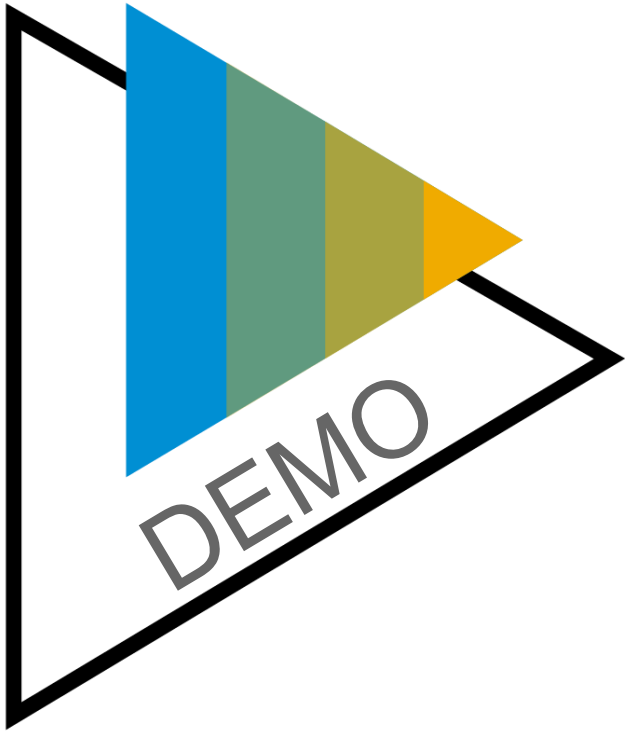


Demo

Demo Scenario

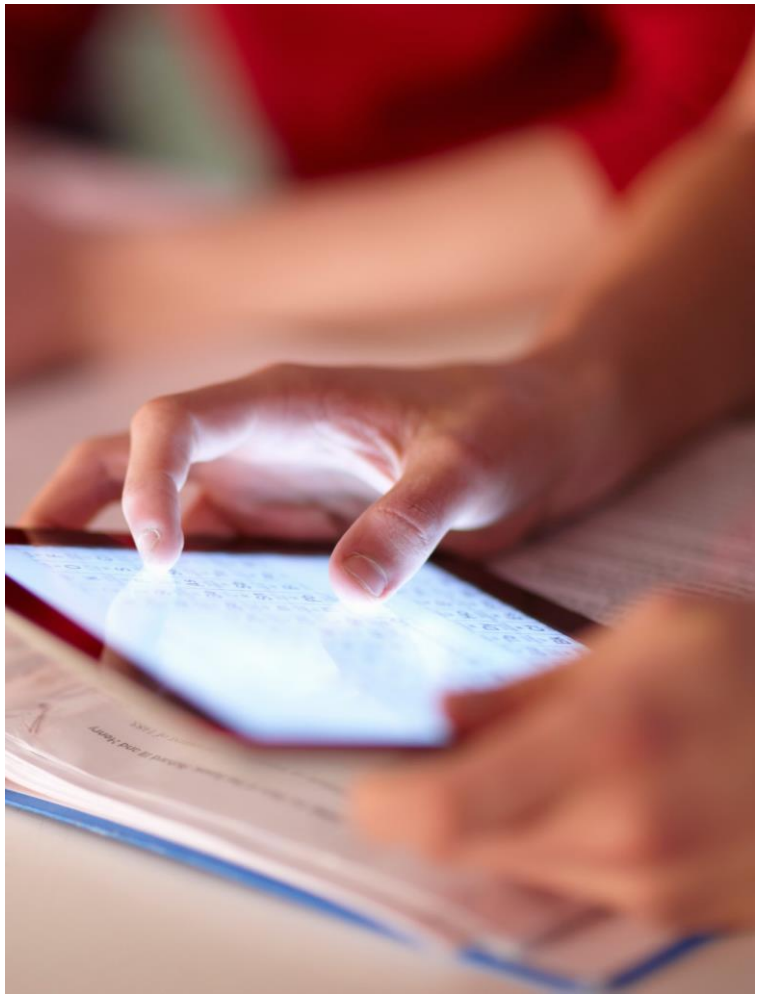
You will build a Node.js application – a Bookshop.
Its data model consists of

- Books
- Authors
- Orders



What you've learned in this unit

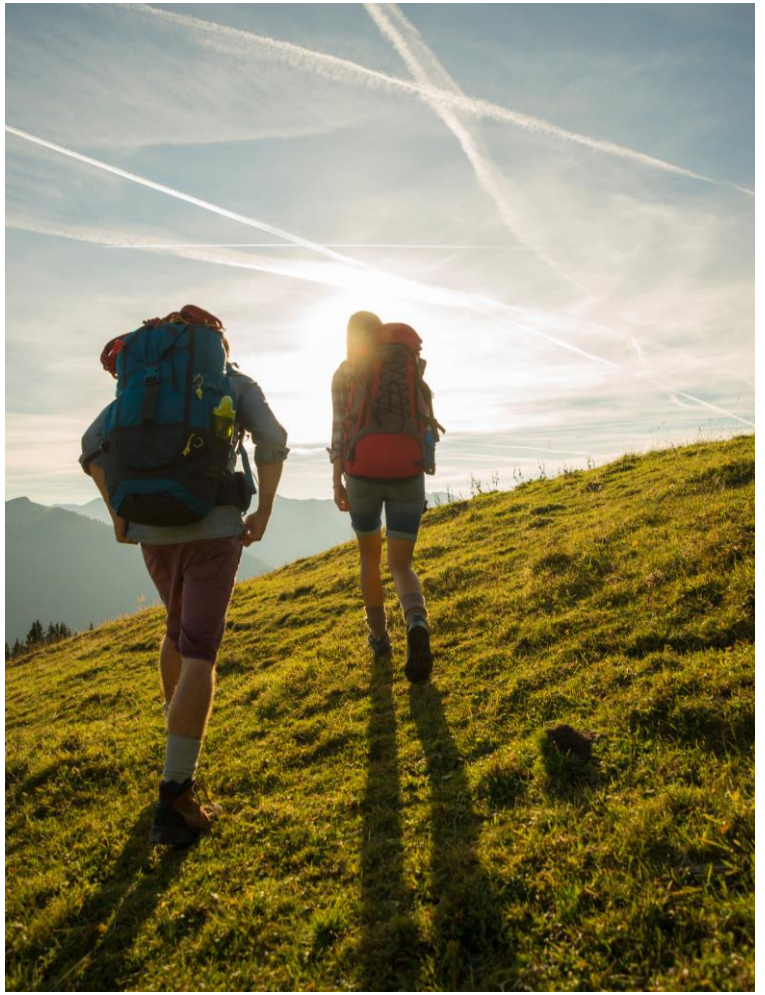
- How to create a Node.js application using the SAP Cloud Application Programming Model
- How to run a service with the CDS command line tool
- Recommended path for (local) development using SQLite and "cds watch"



Further reading

Additional Material

- Official documentation
- Getting Started Guide
- Blogs Cloud Application Programming Model
- Starter Scenario
- TechEd Replay



Thank you.

Contact information:

open@sap.com

Follow all of SAP

www.sap.com/contactsap

THE BEST RUN SAP®