BATCH : B107 AWS-DevOps

LESSON : **AWS**

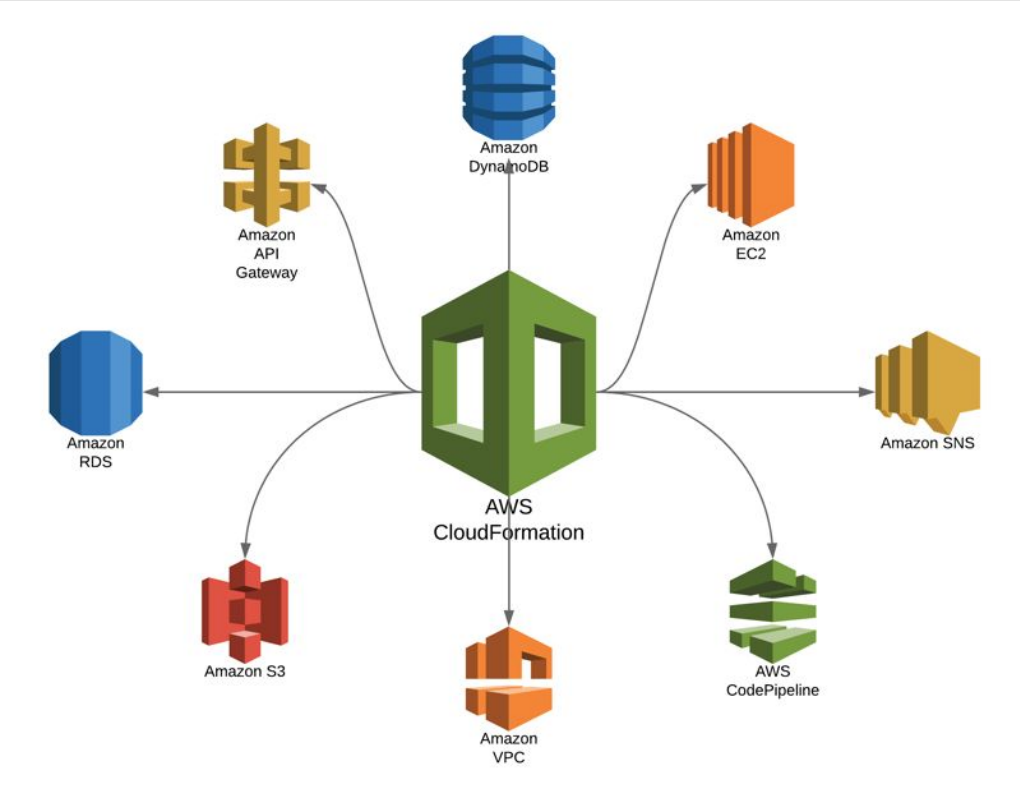
DATE : 04.02.2023

SUBJECT : **AWS-Cloudformation**

ZOOM GİRİŞLERİNİZİ LÜTFEN **LMS** SİSTEMİ ÜZERİNDEN YAPINIZ

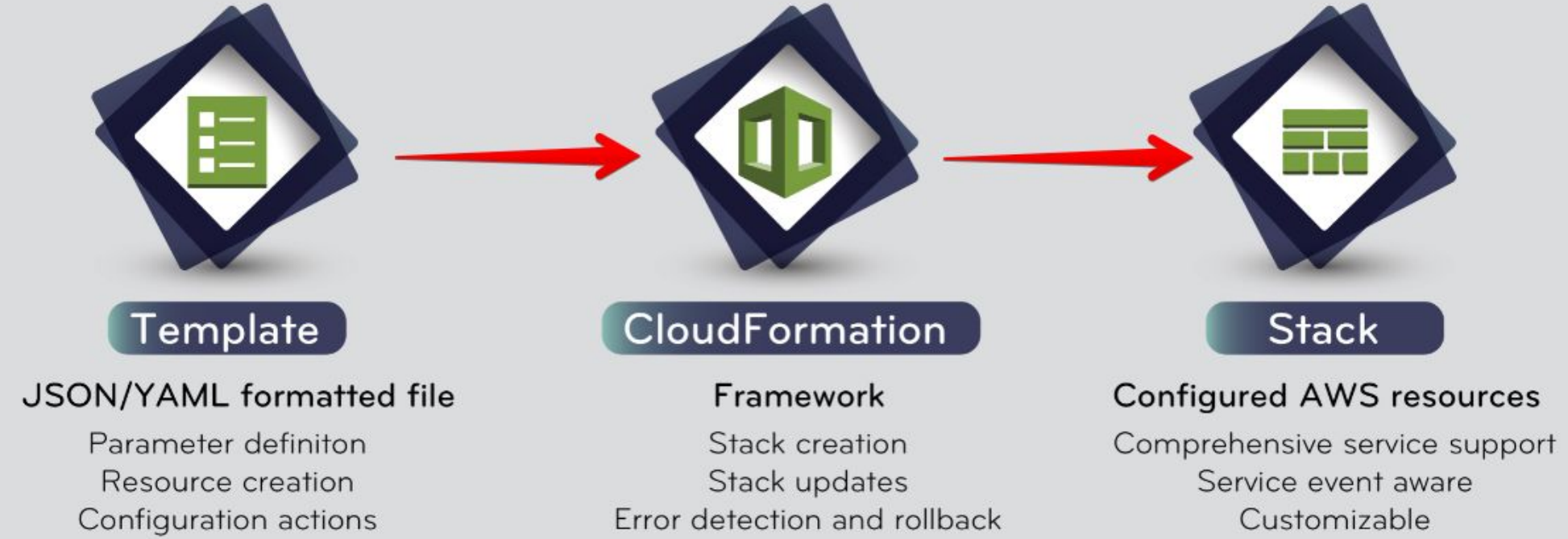
AWS Cloudformation

What is Cloudformation?



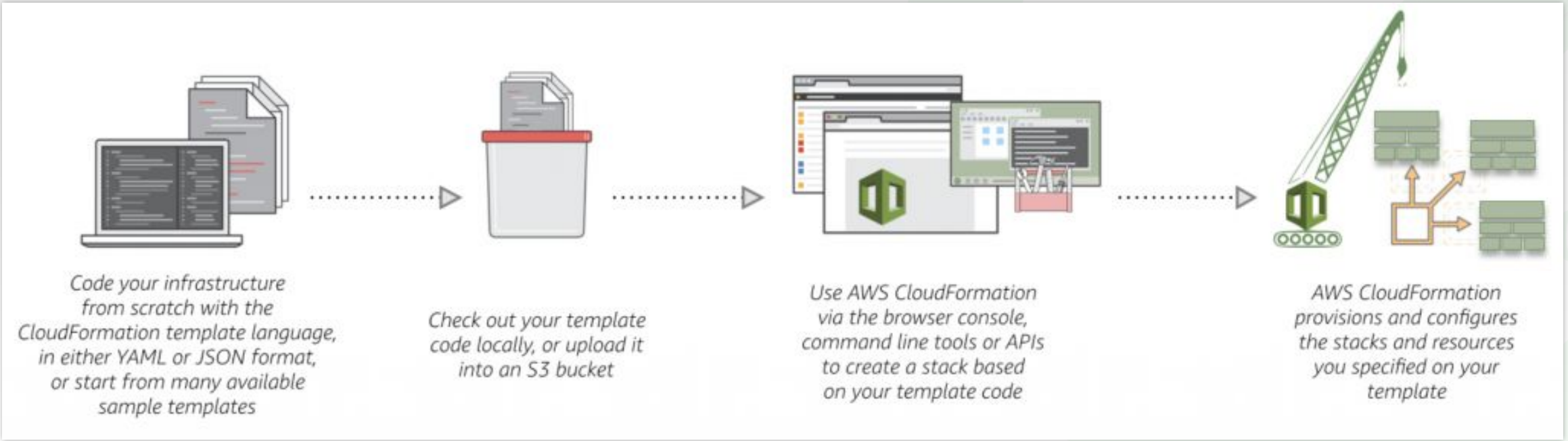
- CloudFormation is an AWS service which enables you to create, manage, configure, replicate and delete AWS resources easily and rapidly using templates, formatted text files in JSON or YAML.

Stacks and Templates



- **Stacks and templates** are the main components of AWS CloudFormation. A template is a JSON or YAML formatted text file which you specify the AWS resources you want to create. Templates can have the extensions .json, .yaml, .template, or .txt
- Declarative programming

How Does AWS Cloudformation Work?



Code your infrastructure from scratch with the CloudFormation template language, in either YAML or JSON format, or start from many available sample templates

Check out your template code locally, or upload it into an S3 bucket

Use AWS CloudFormation via the browser console, command line tools or APIs to create a stack based on your template code

AWS CloudFormation provisions and configures the stacks and resources you specified on your template

Accessing and Pricing

Accessing:

- AWS enables you to access CloudFormation via:

AWS Management Console: You can use Management Console to easily access CloudFormation like many other services.

AWS Command Line Interface: You can also use CLI to access CloudFormation.

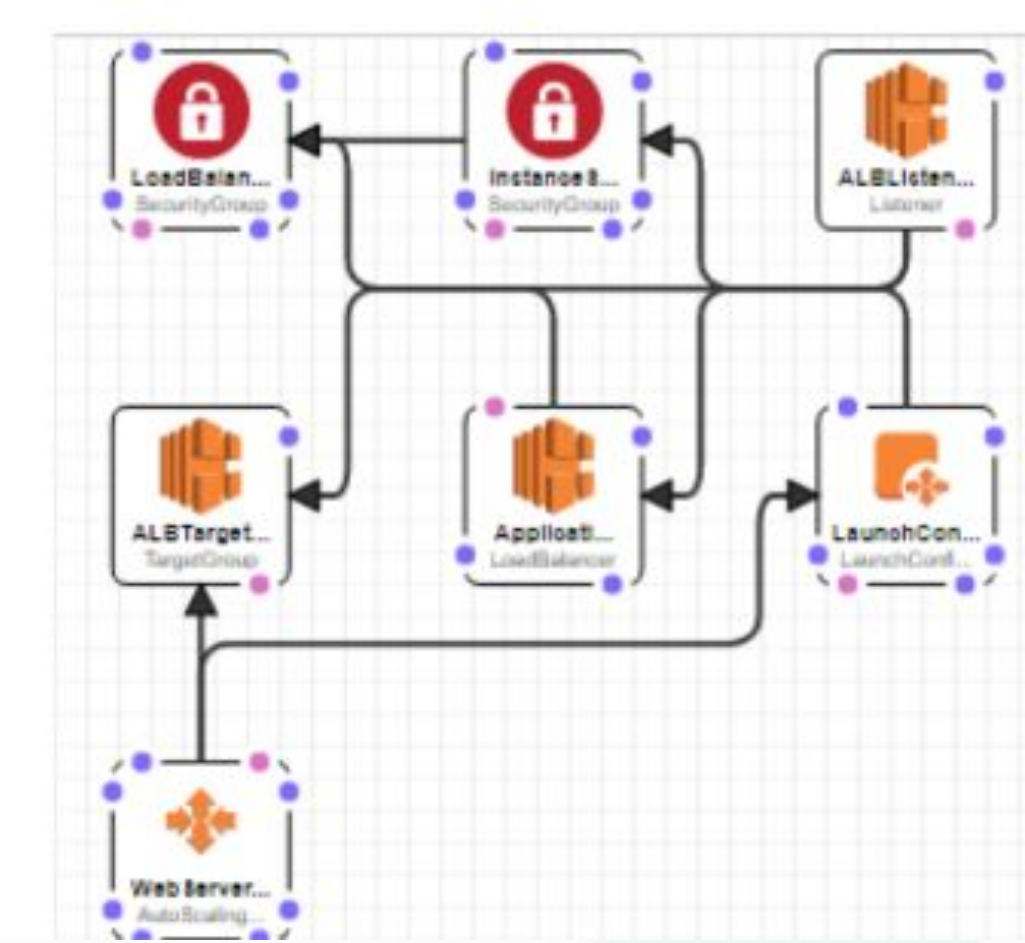
CloudFormation API: AWS supports accessing CloudFormation via API. Consult AWS CloudFormation API Reference.

Pricing:

YOU PAY THE RESOURCES YOU USE

What is Stack?

- A Stack is a single unit composed of the AWS resources provisioned by Cloudformation.



TEMPLATES

- Template Version
- Description
- Metadata
- Parameters
- Mappings
- Conditions
- Transform
- Resources
- Outputs

Template Structure

```
AWSTemplateFormatVersion: 2010-09-09
Description: CloudFormation template for s3 bucket
Resources:
  S3Bucket:
    DeletionPolicy: Retain
    Type: 'AWS::S3::Bucket'
    Description: Creating Amazon S3 bucket from CloudFormation
    Properties:
      AccessControl: Private
      PublicAccessBlockConfiguration:
        BlockPublicAcls: true
        BlockPublicPolicy: true
        IgnorePublicAcls: true
        RestrictPublicBuckets: true
      BucketEncryption:
        ServerSideEncryptionConfiguration:
          - ServerSideEncryptionByDefault:
              SSEAlgorithm: AES256
      VersioningConfiguration:
        Status: Enabled
Outputs:
  S3Bucket:
    Description: Bucket Created using this template.
    Value: !Ref S3Bucket
```

Format Version	Identifies the capabilities of the template
MetaData	Additional information about the template
Description	A description of what this template does
Parameters	Values to pass to your template at runtime
Mappings	A lookup table, maps keys to values so you can change your values
Conditions	Whether resources are created or properties are assigned
Transform	Applies macros
Resources	A resource you want to create
Outputs	Values that are returned

YAML

What is YAML ?

```
# What does YAML mean?
YAML:
    - Y: YAML
    - A: Ain't
    - M: Markup
    - L: Language
```

YAML

What is YAML ?

- Its goal was another markup language, but YAML was filling the role of a data serialization language later, so it was called YAML Ain't Markup Language, which is a recursive acronym.

https://yaml.org/

Attributes of YAML

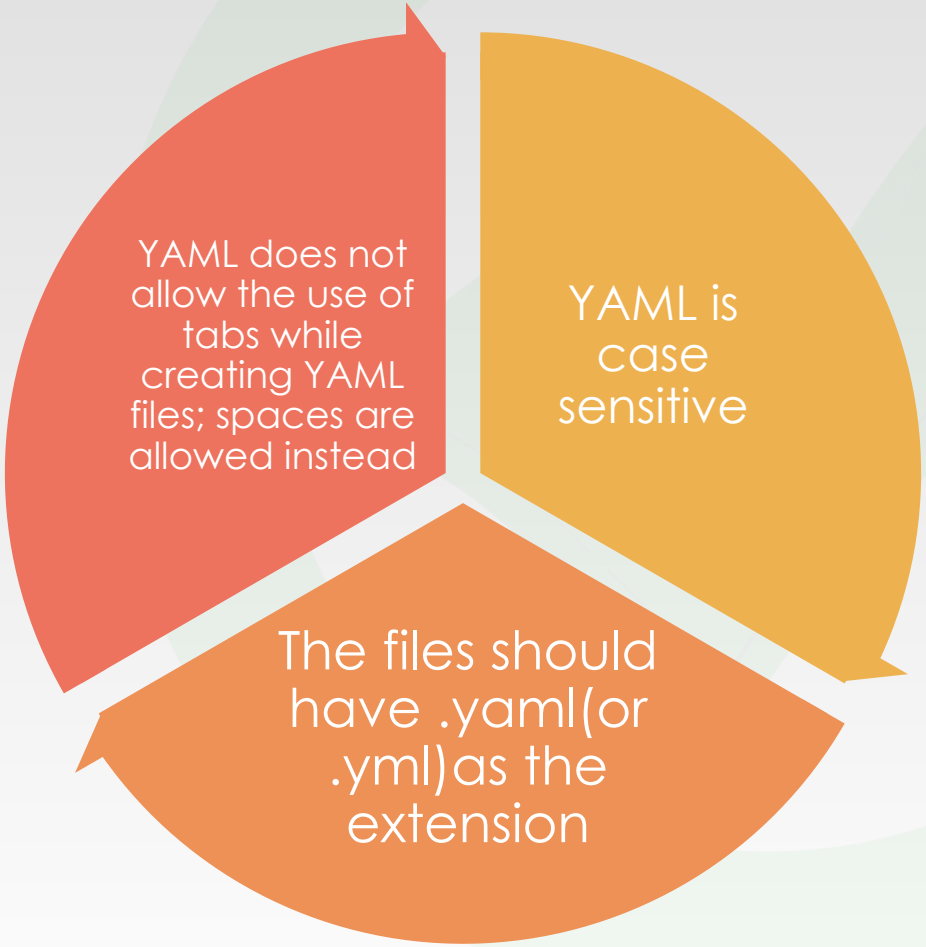
- YAML is human readable.
- YAML is portable.
- YAML works across multiple programming languages easily.
- YAML matches the native data of structures, to agile languages.
- YAML is consistent and is able to support generic tools.
- YAML supports one-pass processing. When a programming language looks at YAML file, it only needs to go through it once to complete its task.
- YAML is expressive easy to extend and adapt.
- YAML is easy to implement and use.

Basics of YAML

- When you are creating a file in YAML, you should remember the following basic rules

YAML does not allow the use of tabs. Spaces are used instead of tabs because tabs are not universally supported.

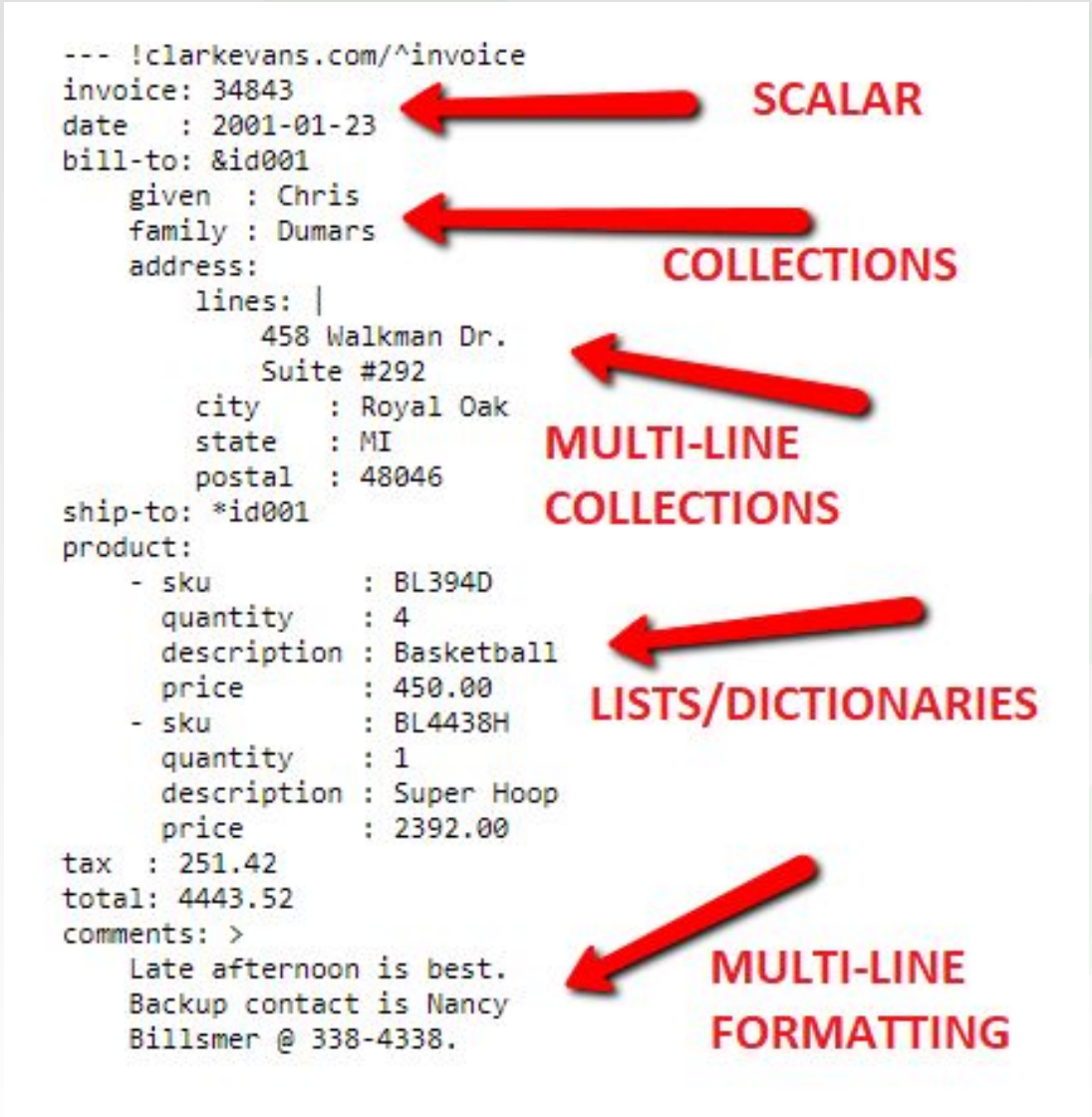
https://yaml.org/



Basics of YAML

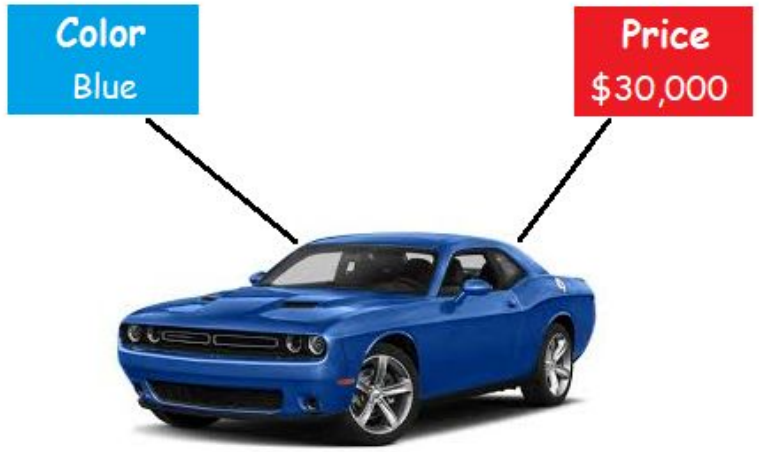
- The key-value is YAML's basic building block. The key is always a string. The value is a scalar so that it can be any data type.

  https://yaml.org/



What is JSON?

- **J**ava**S**cript **O**bject **N**otation



YAML

```yaml
car:
    color: blue
    price: $30,000
```

JSON

```json
{
    "car": {
        "color": "blue",
        "price": "$30,000"
    }
}
```

JSON vs YAML

Data Serialization Languages		
JSON	YAML	XML
JavaScript Object Notation	YAML Ain't Markup Language	eXtensible Markup Language
Data Interchange	Data Interchange	Markup Language
2002	2006	1996
Easy to read	Easier to read	A little complex
Fast	Fast	Slow
Map Structure	Map Structure	Tree Structure
.json	.yaml	.xml

- JSON is a subset of the JavaScript object notation syntax.
- JSON data is stored in name/value pairs.
- JSON records are separated by commas.
- JSON field names & strings are wrapped by double quotes.

YAML

```yaml
apis:
  - name: login
    port: 8080
  - name: profile
    port: 8090
```

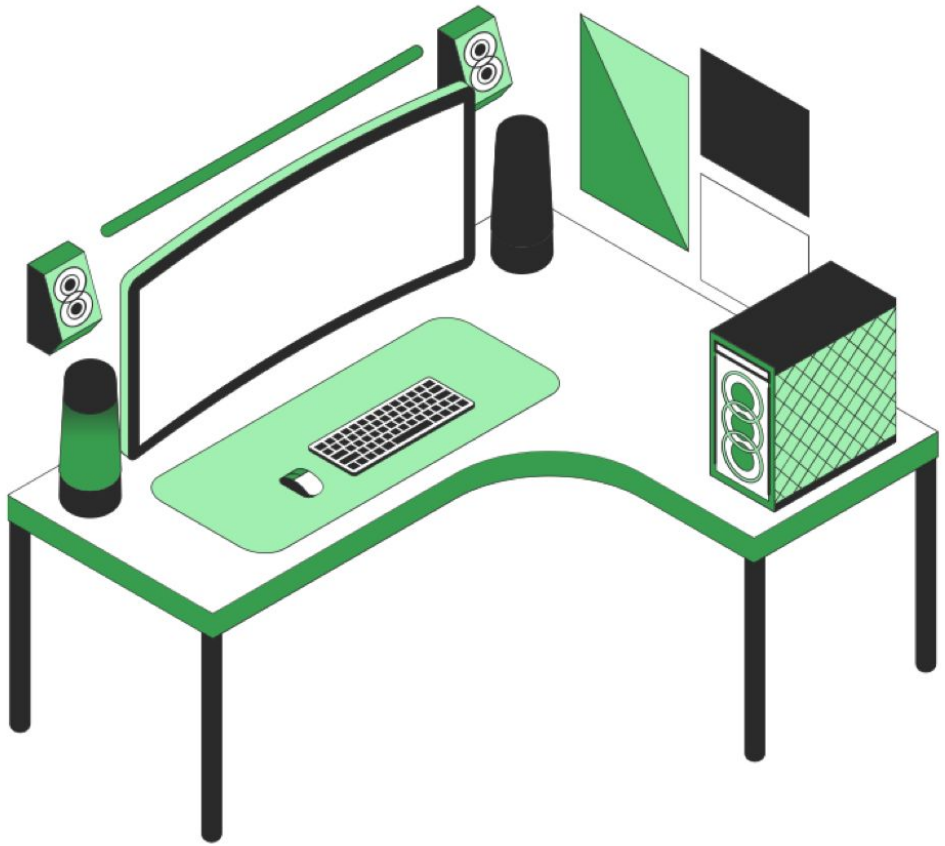
XML

```xml
<apis>
    <api>
        <name>login</name>
        <port>8080</port>
    </api>
    <api>
        <name>profile</name>
        <port>8090</port>
    </api>
</apis>
```

JSON

```json
{
  "apis": [
    {
      "name": "login",
      "port": 8080
    },
    {
      "name": "profile",
      "port": 8090
    }
  ]
}
```

- YAML stands for ain't markup language and is a superset of JSON
- YAML files begin with '- - -', marking the start of the document.
- YAML documents end with '...' but it's optional.
- YAML key value pairs are separated by colon.
- YAML lists begin with a hyphen(-).



Do you have any questions?

Send it to us! We hope you learned something new.