Project work (group) - AWS Lambda

1. Create a simple, Maven project with Java 11
2. Add Maven dependencies:

- aws-lambda-java-core
- aws-lambda-java-events

Remember to set **proper source and target** in maven compiler plugin which should be 11.

Check out the sample project for the reference.

3. Add class that will implement RequestHandler interface from was lambda java core library.

Check out the sample project for the reference.

4. Implement handleRequest method (input type: S3Event, output type: String). Inside this method implement given behavior: r**ead object bucket name and url decoded key.** Log those informations.

You can check out this tutorial to find a way how you can log informations in AWS Lambda functions: https://docs.aws.amazon.com/lambda/latest/dg/java-logging.html.

You can use **LambdaLogger** (simple), or **Log4J/SLF4J**.

Check out the sample project for the reference.

5. Build your project to .jar file with Maven

6. Go to AWS Console and create S3 Bucket

Give it a name, use default settings.

Remember the **region** you chose! You need to select the same region in your Lambda creation.

7. In AWS console create a Lambda function.

In AWS Console select „**Lambda**” service.

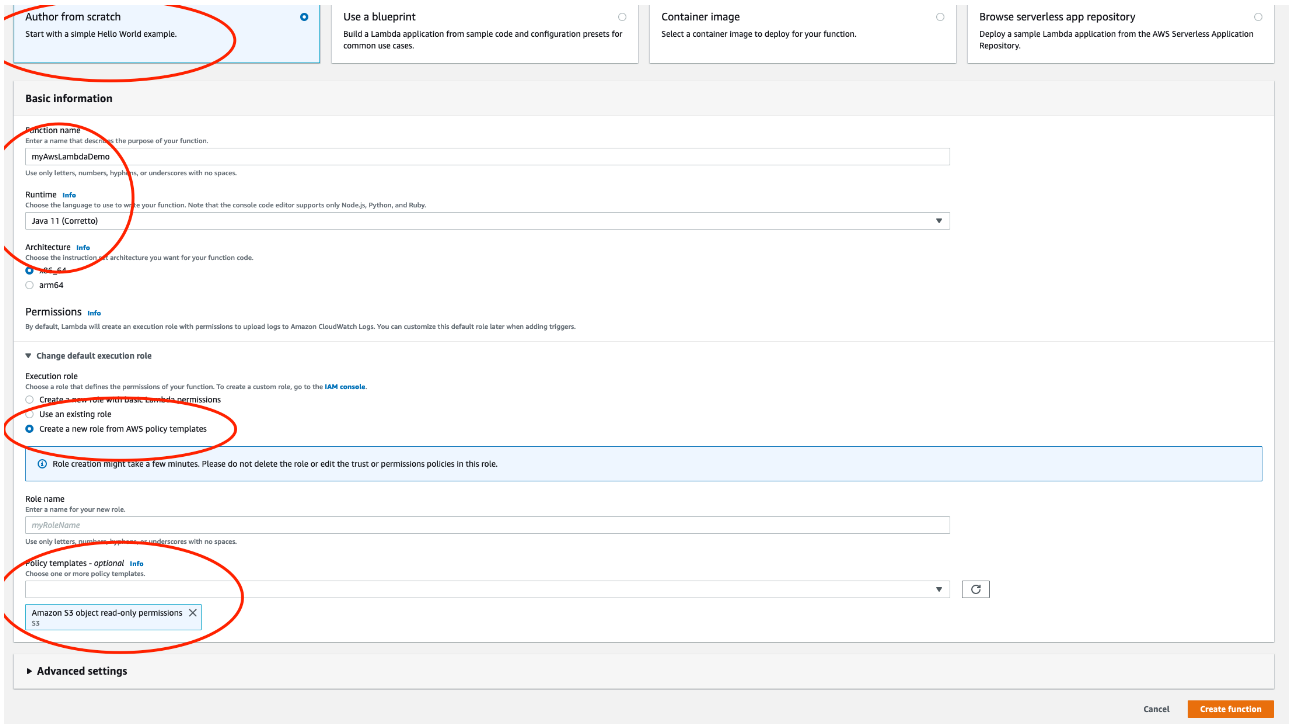
Set a function name, select **Java 11 (Correto)** as runtime and Architecture **„x86_64”**

Then, go permissions and select „**Change default execution role**”.

Your Lambda needs to have a permission that **allows reading from S3**.

So select **„Create a new role from AWS policy templates”**, and give it a name.

Select from Policy templates - **„Amazon S3 object read-only permissions”**.



Author from scratch
Start with a simple Hello World example.

Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

Container image
Select a container image to deploy for your function.

Browse serverless app repository
Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name
Enter a name that describes the purpose of your function.
myAwsLambdaDemo
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Java 11 (Corretto)

Architecture Info
Choose the instruction set architecture you want for your function code.
[✓ x86_64]
arm64

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
Use an existing role
[✓ Create a new role from AWS policy templates]

Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

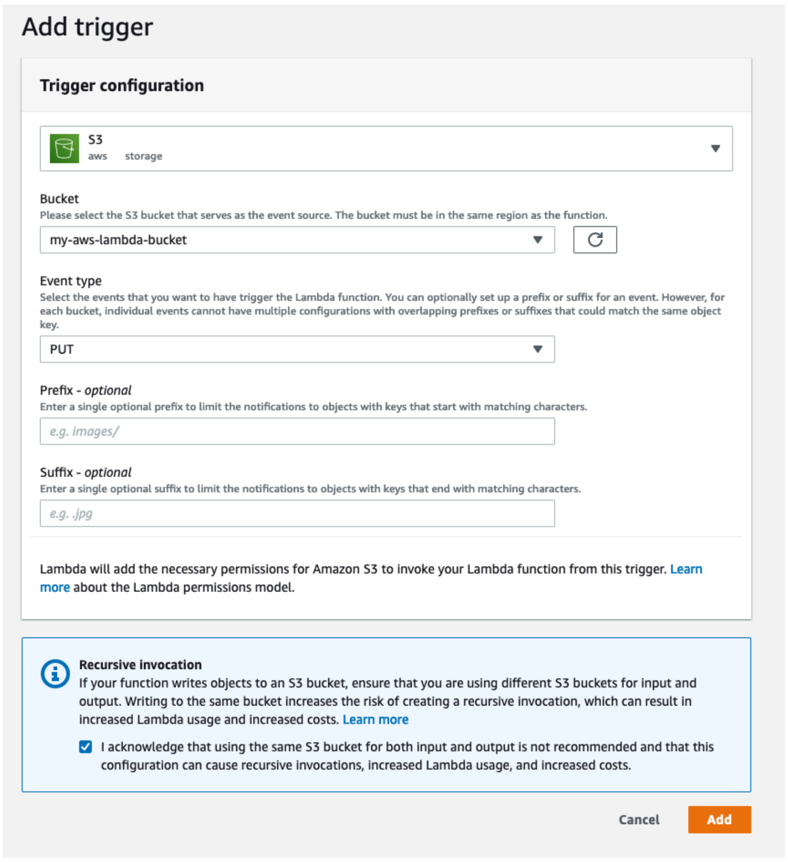
Role name
Enter a name for your new role.
myRoleName
Use only letters, numbers, hyphens, or underscores with no spaces.

Policy templates - *optional* Info
Choose one or more policy templates.
Amazon S3 object read-only permissions
S3

▸ Advanced settings

Click **„Create function”**.

8. Add a trigger to created Lambda function by selecting a bucket created in 5 step. Choose **Event type: PUT** so that your Lambda is triggered when a new file arrives in S3 bucket.



Add trigger

Trigger configuration

S3
aws storage

Bucket
Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.
my-aws-lambda-bucket

Event type
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.
PUT

Prefix - *optional*
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.
e.g. images/

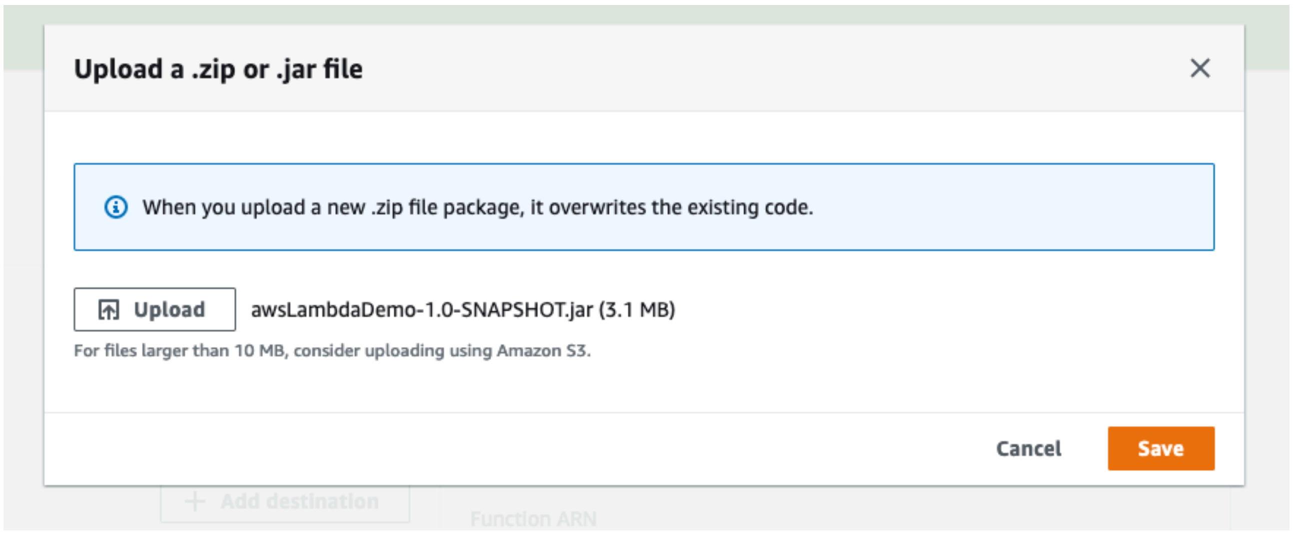
Suffix - *optional*
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.
e.g. .jpg

Lambda will add the necessary permissions for Amazon S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

Recursive invocation
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. Learn more

☑ I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

9. In Lambda configuration go to „Code” tab. In code source select „Upload from” -> .zip or .jar file -> Upload your project that was built in step 5.

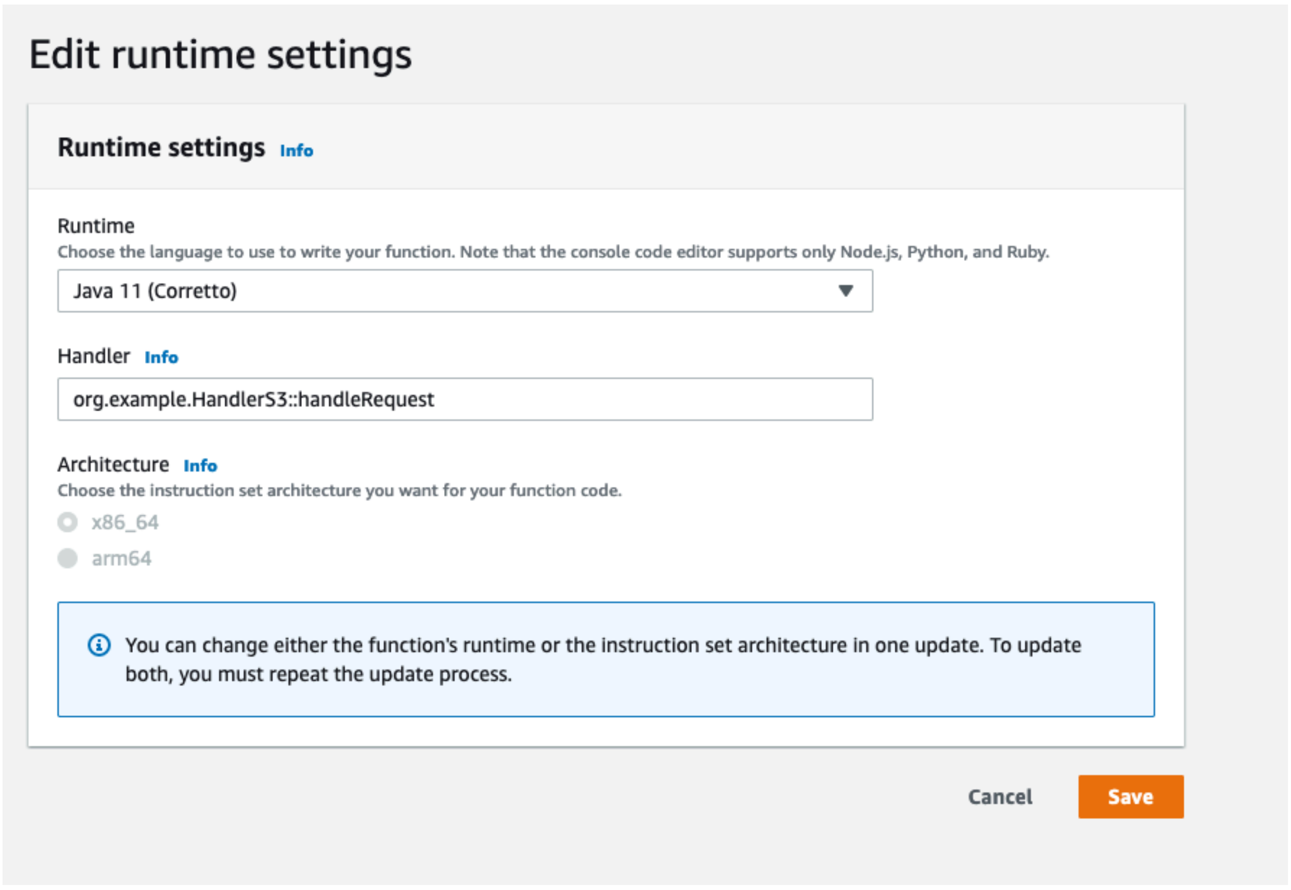


When you upload a new .zip file package, it overwrites the existing code.

awsLambdaDemo-1.0-SNAPSHOT.jar (3.1 MB)

For files larger than 10 MB, consider uploading using Amazon S3.

10. In Lambda configuration in „Code” tab go to „Runtime settings”. Make sure that Handler property is set properly - it should have proper **package and class name**, according to what is in your Java project.



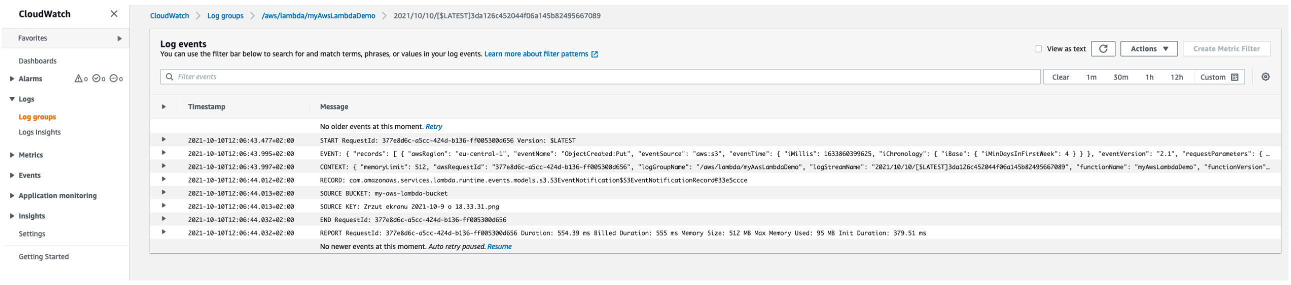
Runtime
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Java 11 (Corretto)

Handler
org.example.HandlerS3::handleRequest

Architecture
Choose the instruction set architecture you want for your function code.
x86_64
arm64

You can change either the function's runtime or the instruction set architecture in one update. To update both, you must repeat the update process.

11. Now **upload** any file into your **S3 bucket**.
12. Go to **„Monitor”** tab in your Lambda configuration. Then open **„View logs in CloudWatch”**.
13. Open recent logs, and see if you have logged details about given S3 object.



Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                      Message
                               No older events at this moment. Retry
2021-10-10T12:06:43.477+02:00  START RequestId: 377e8d6c-a5cc-424d-b136-ff005300d656 Version: $LATEST
2021-10-10T12:06:43.995+02:00  EVENT: { "records": [ { "awsRegion": "eu-central-1", "eventName": "ObjectCreated:Put", "eventSource": "aws:s3", "eventTime": { "iMillis": 1633860399625, "iChronology": { "iBase": { "iMinDaysInFirstWeek": 4 } } }, "eventVersion": "2.1", "requestParameters": { …
2021-10-10T12:06:43.997+02:00  CONTEXT: { "memoryLimit": 512, "awsRequestId": "377e8d6c-a5cc-424d-b136-ff005300d656", "logGroupName": "/aws/lambda/myAwsLambdaDemo", "logStreamName": "2021/10/10/[$LATEST]3da126c452044f06a145b82495667089", "functionName": "myAwsLambdaDemo", "functionVersion"…
2021-10-10T12:06:44.012+02:00  RECORD: com.amazonaws.services.lambda.runtime.events.models.s3.S3EventNotification$S3EventNotificationRecord@33e5ccce
2021-10-10T12:06:44.013+02:00  SOURCE BUCKET: my-aws-lambda-bucket
2021-10-10T12:06:44.013+02:00  SOURCE KEY: Zrzut ekranu 2021-10-9 o 18.33.31.png
2021-10-10T12:06:44.032+02:00  END RequestId: 377e8d6c-a5cc-424d-b136-ff005300d656
2021-10-10T12:06:44.032+02:00  REPORT RequestId: 377e8d6c-a5cc-424d-b136-ff005300d656 Duration: 554.39 ms Billed Duration: 555 ms Memory Size: 512 MB Max Memory Used: 95 MB Init Duration: 379.51 ms
                               No newer events at this moment. Auto retry paused. Resume
```