

M12: Making it easy to develop and support applications with MQ

Matthew Whitehead
mwhitehead@uk.ibm.com



IBM Cloud



Agenda



Why focus on developers?

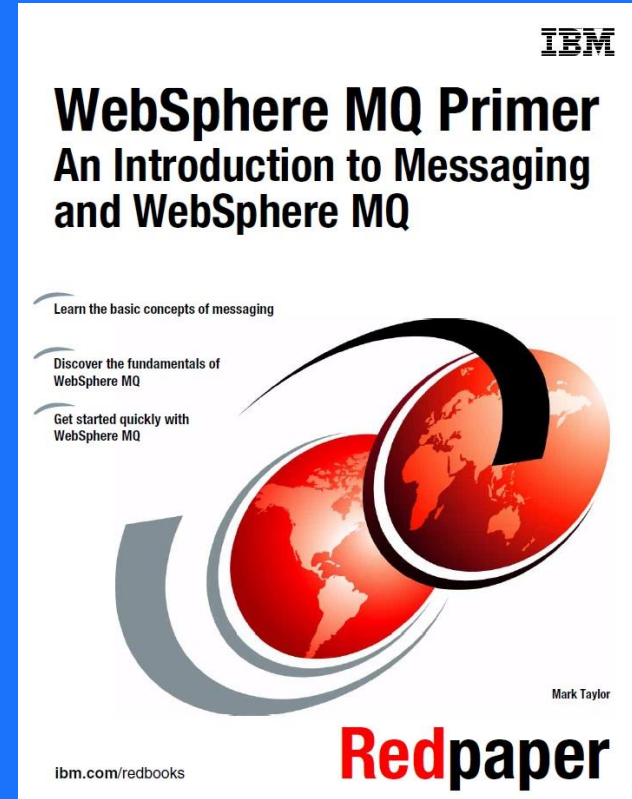
Learn-MQ

The MQ Developer Badge

**Overview of specific things
we built/shipped/created**

- MQ can be hard to get going with for new developers
- How do we make it easier for new application teams to pick up MQ and integrate with their development practices?
- Not always been a product focus – spent more time on administration enablement

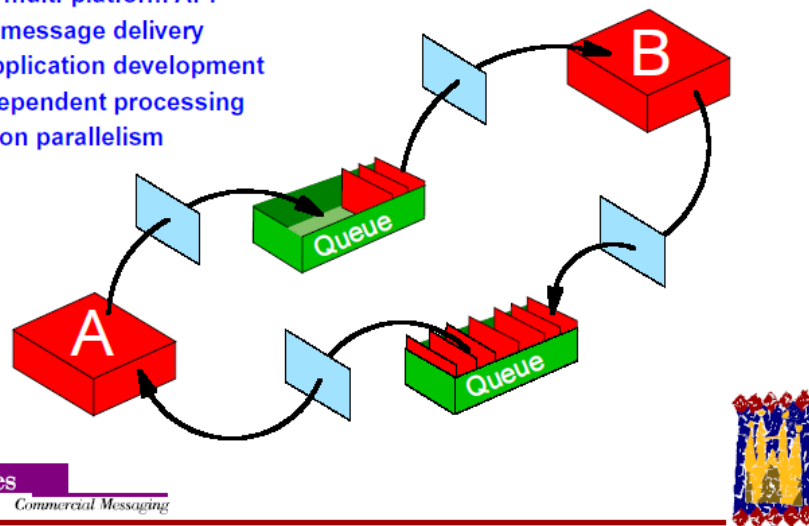
This is what developer engagement used to look like





MQSeries Commercial Messaging

- A single, multi-platform API
- Assured message delivery
- Faster application development
- Time independent processing
- Application parallelism



MQSeries

Commercial Messaging

What's it like now?

LearnMQ

Finding it hard to get developers started with MQ?

Point them to:

developer.ibm.com/
messaging/learn-mq

Totally new to MQ?
Learn the basics

The basics of MQ
With us so far? Great.

Message: packages of data produced and consumed by applications.

Queue: intermediate locations to deliver messages to and store them until they need to be consumed.

Queue manager: virtual MQ engines, the servers that host the queues.

Channels: the way queue managers communicate with each other and with the applications.

MQ networks: loose collections of interconnected queue managers, all working together to deliver messages between applications and locations.

MQ clusters: tight groupings of queue managers, enabling higher levels of messaging availability.

Step-by-step guides to
getting up and running
with MQ

Ready, set, connect!
Connect your first application to a queue manager.

Pick your platform

MQ on Windows
A quick way to learn IBM MQ, set up a queue manager and connect to it from Windows applications. Includes the queue manager console and command line. Includes a sample application that sends and receives messages. Includes a sample application that sends and receives messages. Includes a sample application that sends and receives messages.

MQ on Docker
A quick way to get going with a queue manager and connect to it from Docker applications. Includes the queue manager console and command line. Includes a sample application that sends and receives messages. Includes a sample application that sends and receives messages.

MQ on Cloud
A quick way to get up and running with IBM MQ on cloud. Includes the queue manager console and command line. Includes a sample application that sends and receives messages. Includes a sample application that sends and receives messages.

What you will learn

What you will need

Contents

1. Install Docker

2. Get the MQ in Docker image

3. Run the container from the image

4. Set up your environment

5. Set up your application

6. Set up your database

7. Set up your messaging

8. Set up your messaging

9. Set up your messaging

10. Set up your messaging

Tutorials on building
your applications and
avoiding bad practices

MQ tutorials,
taking you further

Every great achievement starts with a single step. Here's a series of guides to help you with the tasks to master MQ.

Search by:

Protected: Point-to-point with JMS
Write a Java application that sends and receives messages to and from a queue manager. Includes a sample application that sends and receives messages. Includes a sample application that sends and receives messages.

Protected: MQ Essentials
A quick and easy start guide to the fundamental concepts of IBM MQ, including a series of guides to help you with the tasks to master MQ.

Protected: Ready, Set, Connect (Windows)
A quick way to get up and running with IBM MQ on Windows. Includes the queue manager console and command line. Includes a sample application that sends and receives messages. Includes a sample application that sends and receives messages.

Protected: Ready, set, connect (Linux)
A quick way to get up and running with IBM MQ on Linux. Includes the queue manager console and command line. Includes a sample application that sends and receives messages. Includes a sample application that sends and receives messages.

What you will learn

What you will need

Contents

1. Install Docker

2. Get the MQ in Docker image

3. Run the container from the image

4. Set up your environment

5. Set up your application

6. Set up your database

7. Set up your messaging

8. Set up your messaging

9. Set up your messaging

10. Set up your messaging

To enable a user instructed to use MQ for the first time, to go from zero understanding to **running** a sample application in a sandbox environment with a fundamental understanding of MQ concepts in 2 hours

To enable an application developer, instructed to use MQ for the first time, to go from zero understanding to **writing** their first MQ application in the language and environment of their choice within an afternoon

Welcome to messaging, made easy.

IBM MQ is the world's most powerful messaging solution.
LearnMQ will teach you how to master it.



ibm.biz/learn-mq

MQ tutorials, taking you further

Every great achievement starts with a single step.
Here's a set of guided tutorials that provides you with the tools to master MQ.



MQ Essentials (5-10 mins)

Get a queue manager (10-15 mins)

Hello world (10-15 mins)

25 years in 25 minutes!

LearnMQ

Developer Essentials Badge and challenge

Samples and patterns

Welcome to messaging, made easy.

IBM MQ is the world's most powerful messaging solution. LearnMQ will teach you how to master it.

IBM MQ allows developers to focus on business logic and produce code that matters.

What is IBM MQ?

IBM MQ is a robust, reliable and secure messaging solution.

IBM MQ Developer Essentials course

IBM MQ is everywhere. Learn how to use the most powerful messaging solution and create apps that make the world work. Take the course and earn the badge.

Learning objectives:

- Understanding of the IBM MQ messaging concepts
- Creating and configuring a simple queue manager, queue, topic
- Developing a simple point to point JMS application that can connect and interact with the queue manager
- Demonstrating learning and skills by taking on a coding challenge and developing a solution to a problem
- Troubleshooting and debugging simple errors and connectivity issues

Complete the tutorials below to gain the practical skills and knowledge to answer our quiz questions and earn yourself an MQ Developer Essentials badge.

Contents

- Introduction to IBM Messaging**

New to Messaging and IBM MQ?

Welcome. Take a look at this quick introduction.

Then master the basics with the MQ Essentials tutorial. Don't forget to come back when you're done.
- Get up and running with a queue manager**

Stand up a queue manager - MQ server - in Docker. Ready, Set, Connect! See everything you need.

You will be writing an application that needs to connect to a queue manager and interact with its objects (queues, topics, etc.)

Why Docker?

The IBM MQ Advanced for developers in Docker comes pre-configured with some basic MQ objects and a security policy that simplifies getting started.

In a production environment the IBM MQ queue managers will likely have been configured in a way that conforms to organizational standards and policy.

As a developer, you'll likely be provided with a set of connection credentials for the queue managers you'll be using, along with specific queue or topic names.

The default developer configuration in Docker models this experience, so as a dev you can focus on building applications.

Yeah, but I know what I'm doing...

No worries, we don't intend to prevent you from using the instructions to build your solution in a way that better fits your own environment(s). You'll find environment variables available to help customise the code and applications we provide.

Come back when you're done.

Working directory for MQ samples

40 commits | 4 branches | 0 releases | 2 contributors

Branch	Commit	Message	Time
master	40	Initial commit	11 days ago
branch	1	Initial commit	11 days ago
branch	1	Initial commit	10 days ago
branch	1	Initial commit	5 days ago
branch	1	Initial commit	10 days ago

samples

Working directory for MQ samples

MQI Paths

The MQI samples, Node.js, Python, Go, require the MQI Client to have been installed and the paths `MQ_INSTALLATION_PATH` and `QPLD_LIBRARY_PATH` set. If you have installed the MQI client manually, ensure that `MQ_INSTALLATION_PATH` is set to the root directory of your MQI Client installation and `QPLD_LIBRARY_PATH` is set to `MQ_INSTALLATION_PATH\lib64`. Do not install any application requirements until `MQ_INSTALLATION_PATH` and `QPLD_LIBRARY_PATH` are set and exported.

Environment variables

All the samples make use of the same environment variables to define MQ connection settings.

- HOST** - Specify the Host name or IP address of your queue manager
- QMGR** - Set the queue manager name
- PORT** - Listener port for your queue manager
- CHANNEL** - MQ Channel name

Welcome to messaging, made easy.

IBM MQ is the world's most powerful messaging solution. LearnMQ will teach you how to master it.

IBM MQ allows developers to focus on business logic and produce code that matters.

MQ Essentials

Getting started with IBM MQ

1. Message queuing and asynchronous messaging

2. The basics of IBM MQ

3. IBM MQ language support and capabilities

Contents

1. Message queuing and asynchronous messaging

2. The basics of IBM MQ

3. IBM MQ language support and capabilities

MQ with TLS

Getting started with securing messages

1. TLS basics

2. Digital certificates on MQ server

3. CipherSpecs and client set up

MQ with Microclimate

Import and run your first IBM MQ application on Microclimate

1. How to start an IBM MQ Developer Environment on Microclimate

2. Import a microclimate application

3. Running an application

What you will learn

1. Docker

2. Digital certificates on MQ server

3. CipherSpecs and client set up

What you will need

1. Docker

2. Digital certificates on MQ server

3. CipherSpecs and client set up

Contents

1. TLS and MQ

2. Using TLS with MQ in Docker

3. Setting up the server keyStore and certificates

4. MQ queue manager in Docker

5. Setting up the client's trustStore and certificates

6. Setting up the client Docker container

7. Edit, compile and run your application to test the connection

MQ messaging REST API tutorial

Examples for getting started with the IBM MQ messaging REST API

1. Basic MQ messaging REST API usage

2. Swagger in the MQ REST API

3. Go and Node.js examples for using the messaging REST API

What you will learn

1. Basic MQ messaging REST API usage

2. Swagger in the MQ REST API

3. Go and Node.js examples for using the messaging REST API

What you will need

1. IBM MQ queue manager: queue and client app credentials

2. curl

3. Go example

4. Node.js example 3 and example 2

Contents

1. What can you do with the MQ messaging REST API

2. Set up your queue manager

3. Configure the request server

4. curl - put a message

5. curl - get a message

6. MQ REST API discovery with Swagger UI

7. MQ messaging REST API examples

Slow MQ app, lost messages and high CPU? Three easy ways to improve your MQ application

Best practices to make your life easier

1. MQ slow? App keeps crashing? CPU unhappy?

2. MQ running slow? Check how you're connecting

3. MQ slow? App keeps crashing? CPU unhappy?

Contents

1. MQ slow? App keeps crashing? CPU unhappy?

2. MQ running slow? Check how you're connecting

3. MQ slow? App keeps crashing? CPU unhappy?

IBM Learning > MQ Dev badge > Blogs > Downloads > Products > Videos > Docs > Get Help

IBM MQ Developer Essentials course

IBM MQ is everywhere. Learn how to use the most powerful messaging solution and create apps that make the world work. Take the course and earn the badge.



Learning objectives:

1. Understanding of the IBM MQ messaging concepts
2. Creating and configuring a simple queue manager, queue, topic
3. Developing a simple point to point JMS application that can connect and interact with the queue manager
4. Demonstrating learning and skills by taking up a coding challenge and developing a solution to a problem
5. Troubleshooting and debugging simple errors and connectivity issues

Complete the tutorials below to gain the practical skills and knowledge to answer our quiz questions and earn yourself an MQ Developer Essentials badge.

Contents

1. Introduction to IBM Messaging
2. Get up and running with your queue manager
3. Get ready to code in Java
4. Take on the messaging coding challenge
5. Handy tips and tricks
6. Complete the quiz and earn your badge

1 Introduction to IBM Messaging

New to Messaging and IBM MQ?

Welcome. Take a look at this [quick introduction](#).

Then master the basics with the [MQ Essentials tutorial](#). Don't forget to come back when you're done.

2 Get up and running with a queue manager

Stand up a queue manager - MQ server - in Docker. Ready, Set, Connect has everything you need.

You will be writing an application that needs to connect to a queue manager and interact with its objects (queues, topics, etc.)

Why Docker?

The IBM MQ Advanced for developers in Docker comes pre-configured with some basic MQ objects and a security policy that simplifies getting started.

In a production environment the IBM MQ queue managers will likely have been configured in a way that conforms to organisational standards and policy.

As a developer, you'll likely be provided with a set of connection credentials for the queue managers you'll be using, along with specific queue or topic names.

The default developer configuration in Docker models this experience, so as a dev you can focus on building applications.

Yeah, but I know what I'm doing...

No worries, we don't want to prevent you from using the instructions to build your solution in a way that better fits your own environment(s). You'll find environment variables available to help customise the code and applications we provide.

Come back when you're done.

IBM Messaging > LearnMQ > MQ Dev badge > Blogs > Downloads > Products > Videos > Docs > Get Help

Welcome to messaging, made easy.

IBM MQ is the world's most powerful messaging solution. LearnMQ will teach you how to master it.



IBM Messaging > LearnMQ > MQ Dev badge > Blogs > Downloads > Products > Videos > Docs > Get Help

MQ Essentials

Getting started with IBM MQ

IBM Messaging > LearnMQ > MQ Dev badge > Blogs > Downloads > Products > Videos > Docs > Get Help

Ready, set, connect!

Connect a simple MQ application to a queue manager.

Pick your platform

To use free IBM MQ tools and to run a demo application, you need both the queue manager (MQ server) and the demo application (MQ client). In this tutorial you get to play with both. There are many ways to get hold of the MQ server. You can install it locally on various operating systems, run MQ in Docker, or provision a queue manager in the cloud. Each of the tutorials on this page shows one example of getting started with MQ. Click to choose a tutorial: either MQ installation on Windows, MQ in Docker or MQ in IBM Cloud.

MQ on Windows

Install IBM MQ, set up queue manager and run a client demo application, all on one Windows environment.

45 minutes

MQ in Docker

No pre-installed Docker and Docker containers, are with all MQ server manager and one with the MQ client demo application.

30 minutes

MQ on Cloud

Install on IBM MQ cloud manager in IBM cloud and connect to it with a client demo application that is running on Linux, Windows or Mac.

30 minutes

What you will learn

1. How about MQ queue managers
2. A bit about MQ queues
3. The basic point to point messaging

What you will need

1. Docker
2. The latest IBM Docker image from Docker Hub
3. IBM MQ in Docker or point to point demo

Contents

1. Install Docker
2. Get the MQ in Docker image
3. Run the demo application in Docker
4. Put and get messages to and from a queue



"What do I need to start this tutorial?" Just your laptop.



1 Install Docker

If you already have Docker version 17.03, continue to the next section. If not, Docker for your platform, go to Docker Community Edition and install. For example, if you're installing on Ubuntu, instructions are here. If your Docker version is earlier than 17.03, you need to install from source, including the latest Docker or version. Once you've installed Docker, come back to continue with the tutorial.

2 Get the MQ in Docker image

Containers are run from images and images are built from a specification listed in a Dockerfile. We will use a pre-built IBM MQ server image from Docker to use as our point to point container and end up with a working MQ installation and a pre-configured queue manager. The IBM MQ demo client application, we will

IBM Messaging > LearnMQ > MQ Dev badge > Blogs > Downloads > Products > Videos > Docs > Get Help

Point to point with JMS

Write and run your first IBM MQ JMS application

IBM Messaging > LearnMQ > MQ Dev badge > Blogs > Downloads > Products > Videos > Docs > Get Help

What you will learn

IBM Developer

IBM Messaging > LearnMQ > MQ Dev badge > Blogs > Downloads > Products > Videos > Docs > Get Help

MQ Badge developer challenge

Join the dots, show off your messaging skills

What you will learn

1. Publish/subscribe basics
2. Request/response basics
3. How to code an MQ client application
4. How to debug your application
5. Transactional messages
6. Message persistence
7. Security

Contents

1. The challenge
2. Architecture overview
3. Publish/subscribe
4. Request/response
5. Code your application
6. Need to debug?
7. What about security?
8. Message persistence
9. Transactional messages
10. Ready for the quiz?

1 The challenge

Congratulations! You've made it. Your team has taken allocation process. The

I know the basics, but

Don't worry, we have it

OK, so what do I get?

We've already built the

Good news so far, but

You will be coding the

Connect to a queue

Subscribe to the

Send a request to

Get a response

Print out the result

What if I get stuck?

We've included a

Search or jump to... Pull requests Issues Marketplace Explore

ibm-messaging / mq-dev-badge-sample

Watch 3 Star 1 Fork 4

Code Issues Pull requests Projects Wiki Insights

Sample code of the MQ developer essentials badge

18 commits 1 branch 0 releases 3 contributors Apache-2.0

Branch: master New pull request Create new file Upload files Find file Clone or download

robben Merge pull request #4 from chughtai/bm Latest commit 3Feb183 on Nov 22, 2018

DockerFiles Add source files to GenerateEvents.jar 5 months ago

ModelAnswer.com/ibm/mq/demo Remove Transactions from ModelAnswer 3 months ago

ModelAnswerWithDefectToFix.com/ibm/mq/... Remove transactions from answer with defect 3 months ago

TicketReseller.com/ibm/mq/demo Remove Transactions from Skeleton App 3 months ago

LICENSE Initial commit 5 months ago

README.md Update README.md 5 months ago

README.md

We've already built the

Good news so far, but

You will be coding the

Connect to a queue

Subscribe to the

Send a request to

Get a response

Print out the result

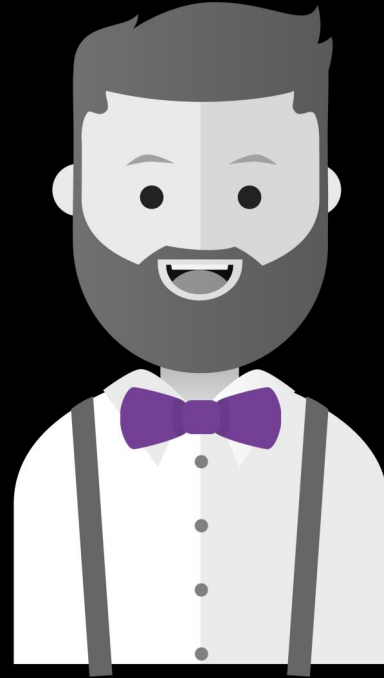
What if I get stuck?

We've included a

Screenshot

This year, MQ is celebrating its 25th birthday and so is Andre, a new application developer.

He needs to deliver a reliable MQ application for his business.

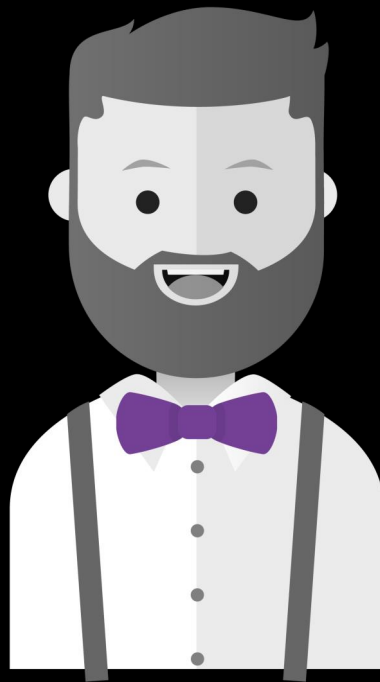


developer: *so what ?*

Just show me an API and I'll code

Working code speaks for itself.

Ready to learn and excited to grow marketable skills.



Andre

Application Developer

My job is to get
code working

I need to deliver a
new application

Skills

Java

C

Ruby

REST

Linux

developer: insight



responsibilities

- app design & logic
- getting new app connected to the MQ Network
- DevOps
- incident resolution
- getting to done!

needs

- find information on MQ
- core concepts of MQ
- build applications with MQ
- some community support
- marketable skills

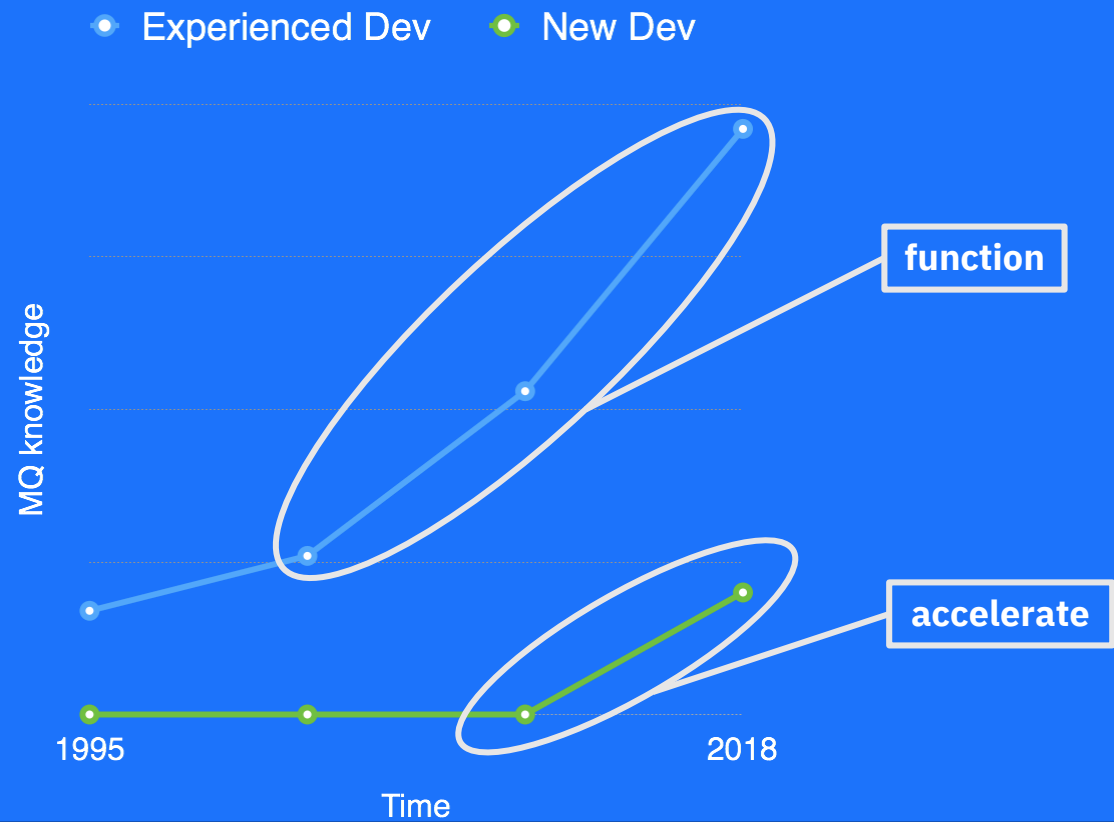
likes

- samples, assets and code to get started (GitHub)
- working in his preferred dev environment
- doing stuff the *right way*
- minimal setup
- enjoying work
- Help when he needs it

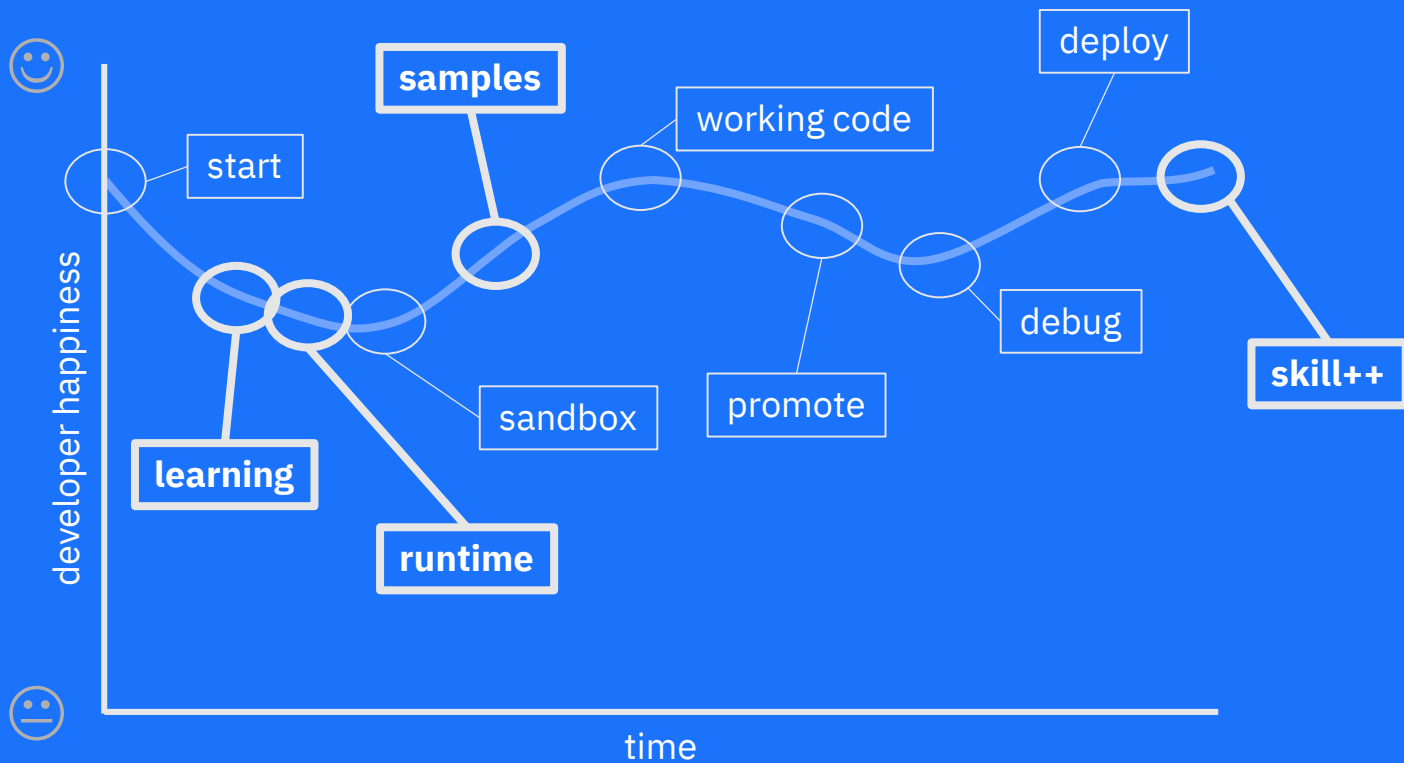
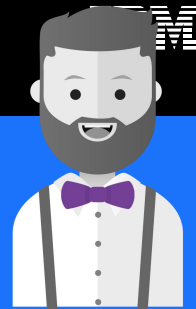


Just one thing,
**He's not used
IBM MQ before**

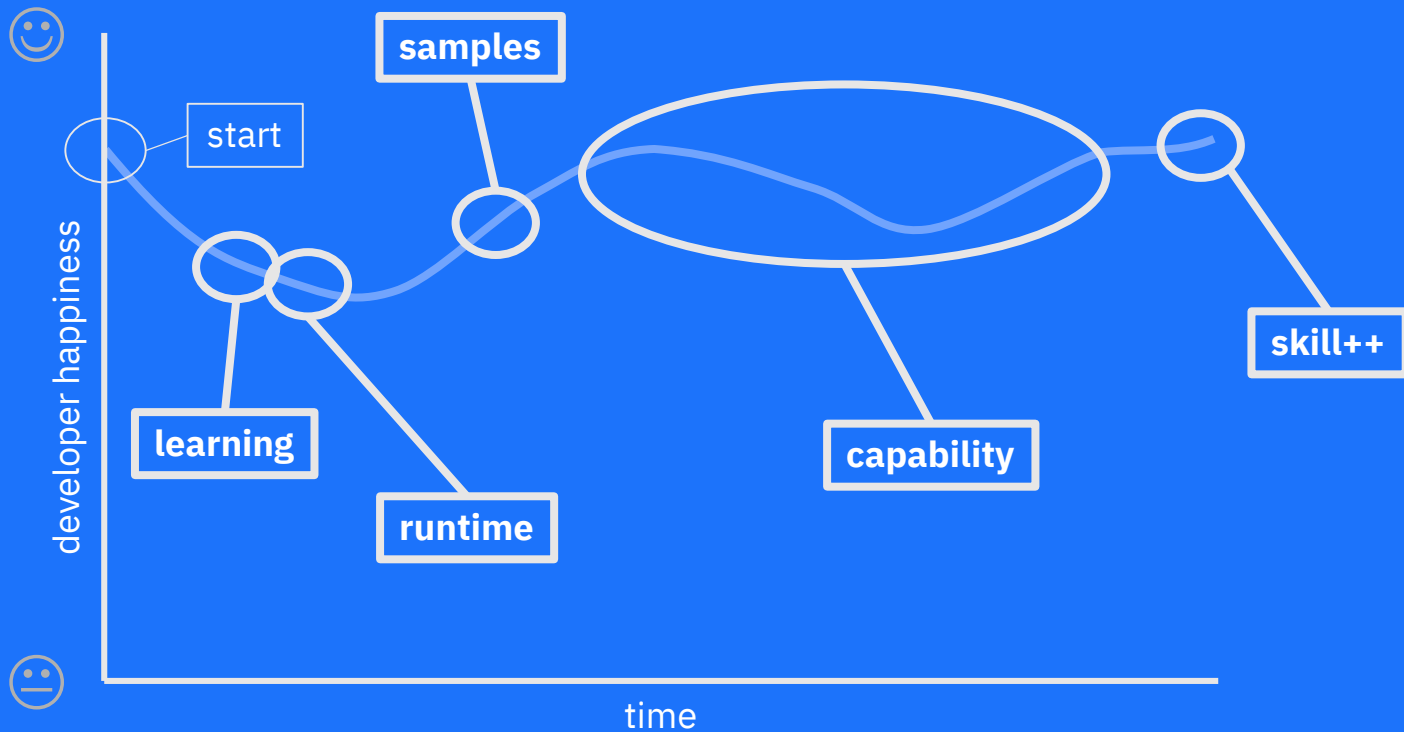
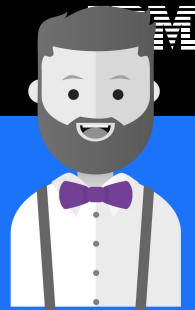
developer: knowledge cycle



dev experience: positive direction



dev experience: positive direction



developer: challenge



*Time to
working code*



*Time to first
Hello World*

More than just APIs: scale, reliability, security and extensibility capabilities are critical.

Need working production ready code to succeed.

IBM MQ provides enterprise grade messaging and is a very marketable skill.

Knowledge Centre

Google

MQSeries.net

Internal Doc

stackoverflow

*>50% coders have fewer than 5 years' professional experience**

*<https://insights.stackoverflow.com/survey/2018>

developerWorks

Demo

Recent activity...

(what have we actually been doing)



APIs &
Protocols

REST

Go

Node.js

Maven

MQLight

Spring

Docker

LTS & CD

Easier ways to get started – Build, IDEs, and tools

- Developers also need easy access to interfaces – no matter their experience
- Do not want to have to install full products
- Many IDEs and build tools integrate with public repositories
- MQ Java interfaces now available from Central Repository (Maven)
 - No need to explicitly install or download
 - Just reference MQ jars in application configuration

Gradle: build.gradle

```
dependencies {  
    compile("com.ibm.mq:com.ibm.mq.allclient:9.1.0.0")  
}
```

Maven: pom.xml

```
<dependency>  
    <groupId>com.ibm.mq</groupId>  
    <artifactId>com.ibm.mq.allclient</artifactId>  
    <version>9.1.0.0</version>  
</dependency>
```

Easier ways to get started – Java

- Many Java developers use Spring
 - Can reduce amount of code needed
 - MQ JMS used with Spring for many years
- Spring Boot & Auto-configure give further code reduction
 - You can get a program running quickly
 - With default capabilities
- MQ now has Spring Boot starter
 - Versions for both Boot 1 and Boot 2
 - Source on github; jar on Maven Central

build.gradle

```
dependencies {  
    compile("com.ibm.mq:mq-jms-spring-boot-starter:+")  
}
```

Easier ways to get started – Runtime Availability

- Docker container with full MQ server function
 - Creates sample objects for developers
 - Queues, topics, userids etc
- See github.com/ibm-messaging/container with several build options
- Redistributable Client packages (Linux, Windows) now easier to access
 - To make it easy for developers to package standalone applications
 - Perhaps in a container
 - No login required to download
 - Now includes SDK (header files/copybooks) to enable application development
 - Simplifies creation of a build environment via Docker containers
- See <https://public.dhe.ibm.com/ibmdl/export/pub/software/websphere/messaging/mqdev/redist/>

Easier ways to get started – Mac

- Mac Client for 9.1.1
- Can be found here: ibm.biz/mq-mac-dev
- Allow Mac users to develop MQ applications
 - Native editor
 - Avoid need for VM just for MQ
- Access C MQI backed language bindings
 - Node.js, Golang, python etc.
- Remote runmqsc
- Local dev using docker container for server side testing
- Not supported for production: dev only.



- Redistributable Client packages now easier to access
 - Make it easy for developers to package standalone applications
 - For example, in a container
 - Windows and Linux
- No SDK install needed to access headers to compile MQ C programs
 - and Cobol and C++
 - A build process might be container-based, creation of container can download/unzip latest SDK

<https://public.dhe.ibm.com/ibmdl/export/pub/software/websphere/messaging/mqdev/redirect/>

Multiple APIs and Protocols

- IBM MQ supports multiple APIs and multiple client protocols. Both proprietary and open.
APIs: **MQI, JMS, MQ Light, REST** ...
Protocols: **MQ, AMQP, MQTT, HTTP**
- These support a wide range of application styles, from the simplest of messaging needs through to the richest
- MQ is the transport; messages produced from any API or protocol can be received by any other API or protocol



APIs

MQI

Exposes the full set of MQ capabilities
Uses the MQ protocol

JMS 2.0

Supports the full JMS API for use in many JSE or JEE environments
Uses the MQ protocol

MQ Light

A simple pub/sub messaging API
Uses the AMQP 1.0 protocol

Protocols

MQ

MQ's highly reliable and performant messaging protocol

MQTT

Supports the open standard MQTT API and protocol
Open source Eclipse Paho clients

AMQP

Support for AMQP 1.0 enables support for open source clients such as Qpid Proton

REST

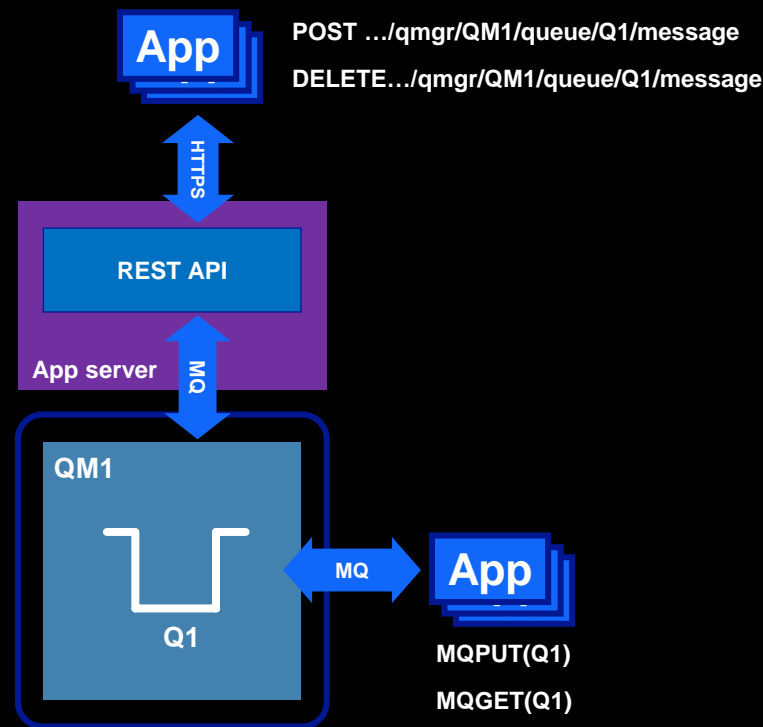
A very simple but secure messaging API over REST

...

REST Messaging



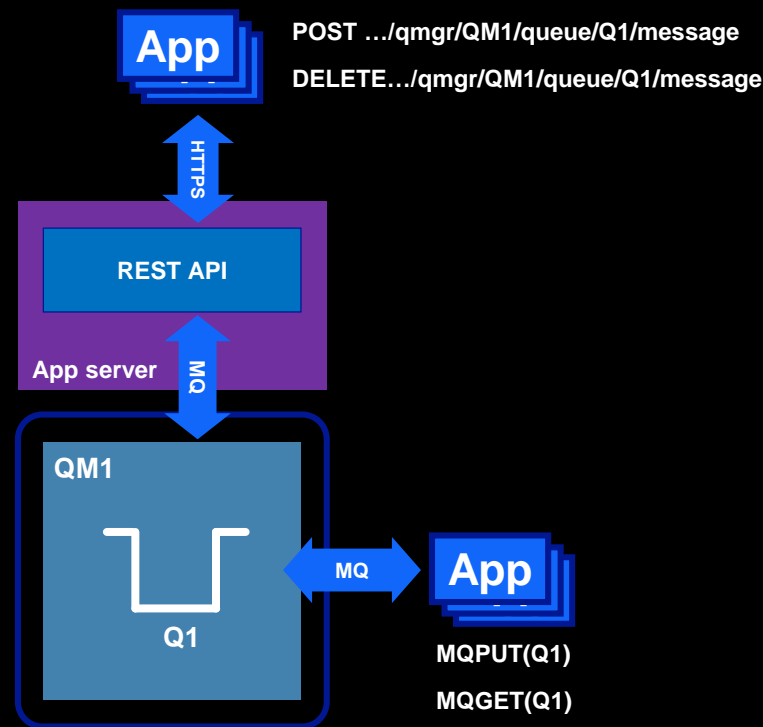
- Many users just want the ***simplest*** way to get messages in and out of an MQ system
- A RESTful API gives you just that. Easily enabling messaging from just about ***any environment*** with no need for an MQ client or platform/language limitations
- MQ support for z/OS Connect now provides a similar mechanism on z/OS.



REST Messaging



- The new HTTP server support in MQ 9.0.x provides the platform for a properly integrated REST API solution
- CD releases start to see a messaging REST API evolve, first with simple synchronous point-to-point support.



New MQI languages

- The MQI API exposes the richest MQ messaging capabilities

Traditionally mostly used by C and COBOL

Other bindings have been available

- New interfaces available for **GO** and **Node.js** designed around full MQI function
- Provided as as-is open source, let us know if there's a need to take these further
- Built on the C MQI library

The screenshot shows the GitHub repository page for 'ibm-messaging/mq-golang'. The repository has 31 commits, 1 branch, 0 releases, 1 contributor, and is licensed under Apache-2.0. The commit history is as follows:

File	Commit Message	Time Ago
cmd	Add the script to startup the collector for the MQ Bridge for Salesforce	7 months ago
ibmmq	Update comments about Windows #cgo compiler directives	4 months ago
mqmetric	Need to check if there really was an error	6 months ago
CLA.md	First code release	a year ago
LICENSE	Initial commit	a year ago
README.md	Update comments about Windows #cgo compiler directives	4 months ago

The screenshot shows the GitHub repository page for 'ibm-messaging/mq-mqi-nodejs'. The repository has 4 commits, 1 branch, 0 releases, 1 contributor, and is licensed under Apache-2.0. The commit history is as follows:

File	Commit Message	Time Ago
lib	Force FIQ option	4 days ago
samples	Force FIQ option	4 days ago
CLA.md	First commit	5 days ago
LICENSE	First commit	5 days ago
README.md	Update link in README for REST API	4 days ago

New MQI languages - examples



Intent to make it easy to write for MQ from newer languages and environments

In a natural style for the language

eg Strings, not padded-fixed-lengths

Helps you to develop apps wherever you need

For **Node.js JavaScript** see github.com/ibm-messaging/mq-mqi-nodejs

Or **require('ibmmq')** from NPM

For **Go** bindings see github.com/ibm-messaging/mq-golang

Both include sample programs to help get started

```
mqmd = new mq.MQMD();
pmo = new mq.MQPMO();
msg = "Hello from Node";

mq.Put(hObj, mqmd, pmo, msg, function(err) {
  if (err) {
    console.log(err);
  }
});
```

```
mqmd := ibmmq.NewMQMD()
pmo := ibmmq.NewMQPMO()
mqmd.Format = "MQSTR"
msg := "Hello from Go"

buffer := []byte(msg)
err = qObject.Put(mqmd, pmo, buffer)
if err != nil {
  fmt.Println(err)
}
```

3rd party contributions



Varying maintenance, capabilities, license

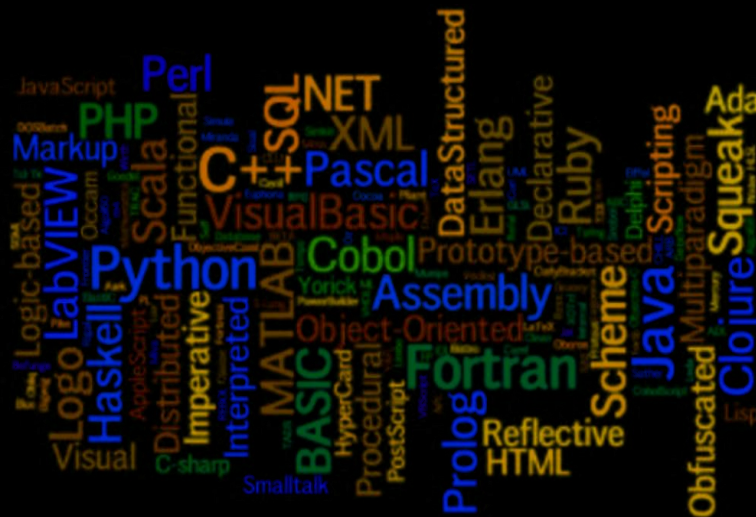
Python: pythonhosted.org/pymqi/

Perl: search.cpan.org/dist/MQSeries/

PHP: pecl.php.net/package/mqseries

Ruby: github.com/reidmorrison/rubywmq

Built on top of C MQI library



MQ Light APIs



MQ Light interfaces in a variety of languages

Client source in GitHub

Integrated with natural public repository

JavaScript: NPM

Java: Maven

Ruby: gem install mqlight

Python: pip install mqlight

.Net blog series at

developer.ibm.com/messaging/2017/11/13/mq-light-messaging-microsoft-net-part-1/

developer.ibm.com/messaging/mq-light

MQ Light Clients

Got the MQ Light Developer Tools? Now choose your preferred programming language to view sample code and client install instructions:

node.js

Java

Ruby

Python

Other

```
# Receive:
require 'mqlight'
client = Mqlight::BlockingClient.new('amqp://localhost')
client.subscribe('news/technology')
delivery = client.receive('news/technology')
puts delivery.data

# Send:
require 'mqlight'
client = Mqlight::BlockingClient.new('amqp://localhost')
client.send('news/technology', 'Hello World!')
```

- Docker container with full MQ server function
 - Creates sample objects for developers
 - Queues, topics, userids etc

- See <https://hub.docker.com/r/ibmcom/mq/>

- Redistributable Client packages now easier to access

- To make it easy for developers to package standalone applications
 - Perhaps in a container

- public.dhe.ibm.com/ibmdl/export/pub/software/websphere/messaging/mqdev/redis

Thank you



- IBM and the IBM logo are trademarks of International Business Machines Corporation, registered in many jurisdictions. Other marks may be trademarks or registered trademarks of their respective owners.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
- Other company, product and service names may be trademarks, registered marks or service marks of their respective owners.
- References in this publication to IBM products and services do not imply that IBM intends to make them available in all countries in which IBM operates.

- THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY.
- WHILST EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.
- IN ADDITION, THIS INFORMATION IS BASED ON IBM’S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE.
- IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION.
- NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, OR SHALL HAVE THE EFFECT OF:
 - CREATING ANY WARRANTY OR REPRESENTATION FROM IBM (OR ITS AFFILIATES OR ITS OR THEIR SUPPLIERS AND/OR LICENSORS); OR
 - ALTERING THE TERMS AND CONDITIONS OF THE APPLICABLE LICENSE AGREEMENT GOVERNING THE USE OF IBM SOFTWARE.