

Introduction to the IBM MQ Console and REST API

Gwydion Tudur (gtudur1@uk.ibm.com)

IBM MQ

November 2018

Session JJ

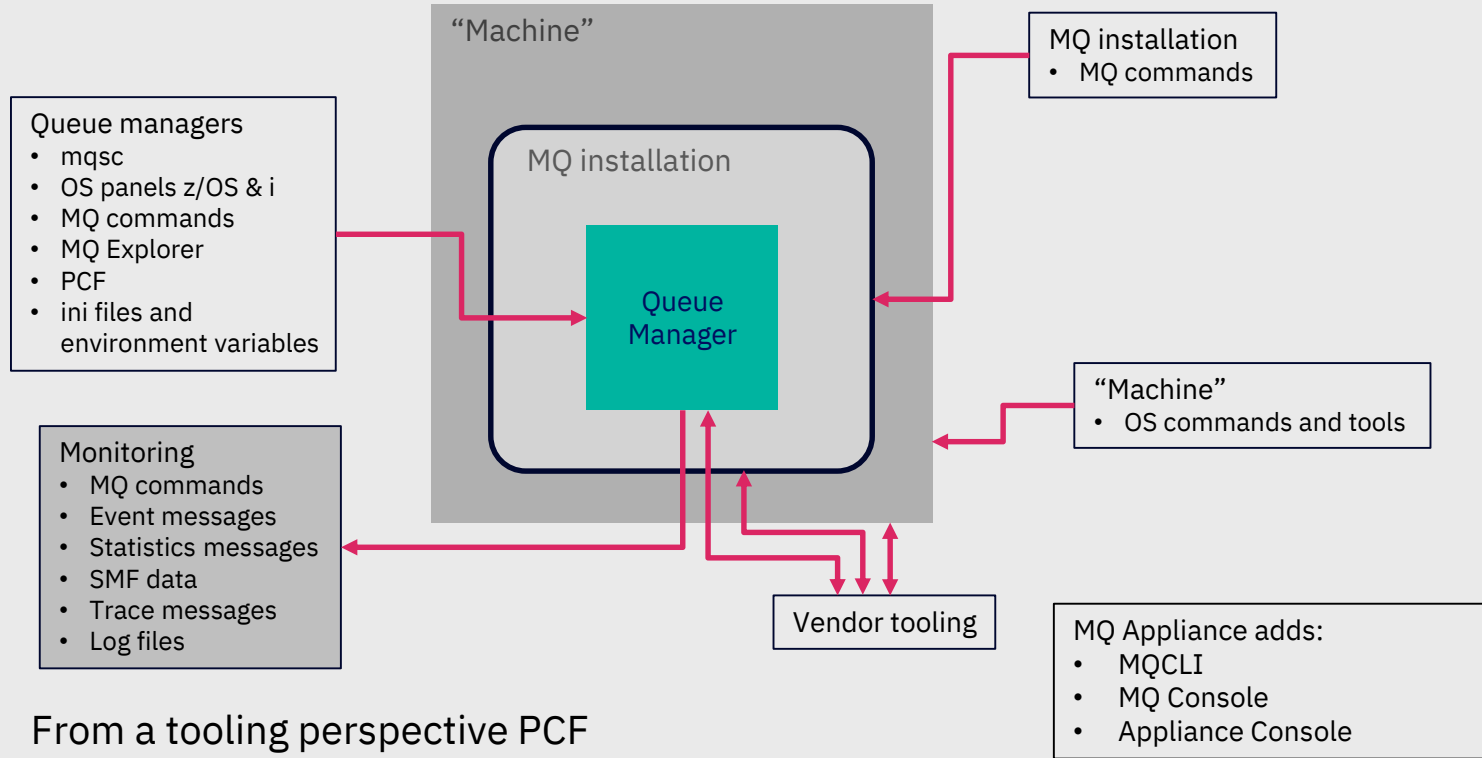


Agenda

- Current options, and why we need something else
- The mqweb server
- The MQ administrative REST API
- Examples
- API Discovery
- Security
- The MQ Console

Current options, and
why we need
something else

Administering software MQ



Why we need more

While PCF is very powerful, it is not that easy to use

- Requires an MQ client, and a supported programming language
- Binary format
- Multiple messages generated per request
- There are tools to make this easier

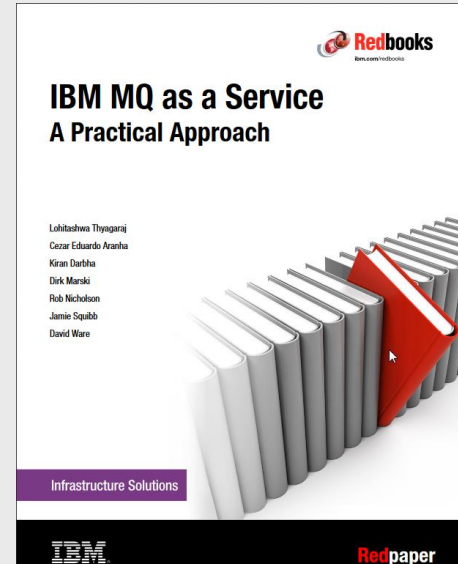
```
**** Message ****
length = 724 of 724 bytes
00000000: 080A 4103 0000 0000 5744 5220 0200 0000  '..A....WDR ....'
00000010: 8800 0000 6700 0000 514D 4752 315F 3230  '...g...QMGR1_20'
00000020: 3135 2D31 302D 3239 5F30 392E 3431 2E31  '15-10-29_09.41.1'
00000030: 3620 2020 2020 2020 2020 2020 2020 2020  '6'
00000040: 2020 2020 2020 2020 514D 4752 3120 2020  ' QMGR1'
00000050: 2020 2020 2020 2020 2020 2020 2020 2020  '
00000060: 2020 2020 2020 2020 2020 2020 2020 2020  '
00000070: 2020 2020 2020 2020 0000 0000 0000 0000  '
00000080: 58CA 0000 0000 0000 0000 0000 0000 0000  'X.....'
00000090: 644E 4656 2116 4656 3230 3135 2D31 302D  'cNFV1_FV2015-10-'
000000A0: 3239 2020 0000 0000 3039 2E34 312E 3233  '29 ...09.41.23'
000000B0: 0100 0000 4D51 4D4D 0000 0000 3038 3030  '...MQMM...0800'
000000C0: 3030 3034 0000 0000 434C 5553 5445 5231  '0004...CLUSTER1'
000000D0: 2E51 4D47 5231 2020 2020 2020 0B00 0000  ' QMGR1 ....'
000000E0: 0800 0000 0200 0000 2020 2020 2020 2020  '
000000F0: 2020 2020 2020 2020 2020 2020 2020 2020  '

```

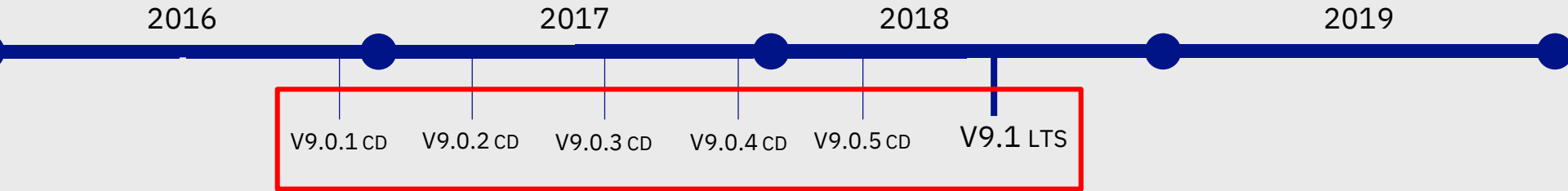
There is a growing need for the ability to administer MQ from

- Any environment
- Any programming language
- By users who are not expert in MQ

Lots of customers are writing self-service web-portals for managing their infrastructure, including MQ



Overview



MQ 9.0.1 CD added support for a number of HTTP-based administration capabilities

- Focus on low barrier to entry and ease of use
- MQ Console – a web-browser based graphical administration tool
- MQ REST API – a programmatic administration API
 - Enhanced further during CD deliverables 9.0.2, 9.0.3, 9.0.4, and 9.0.5

MQ 9.0.4 CD added support for REST messaging

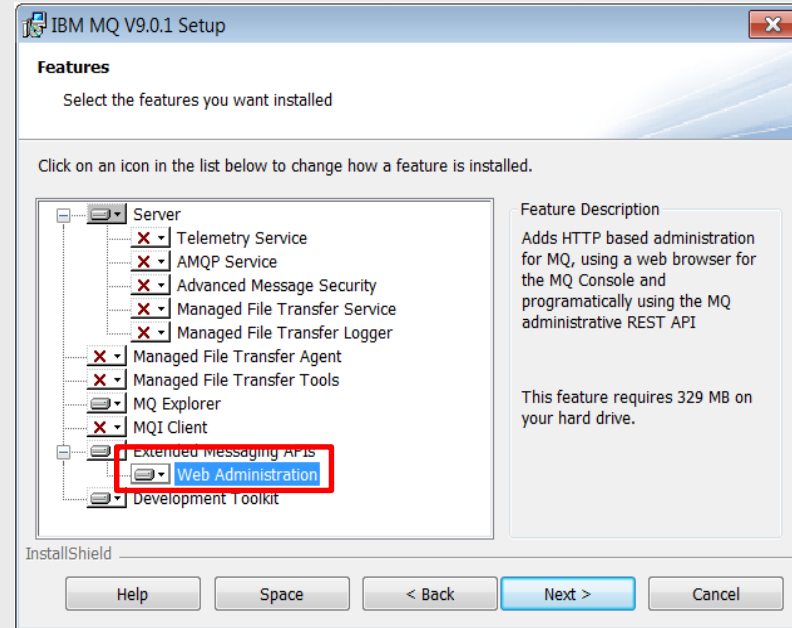
- Basic point to point messaging

All consolidated into the 9.1 LTS release

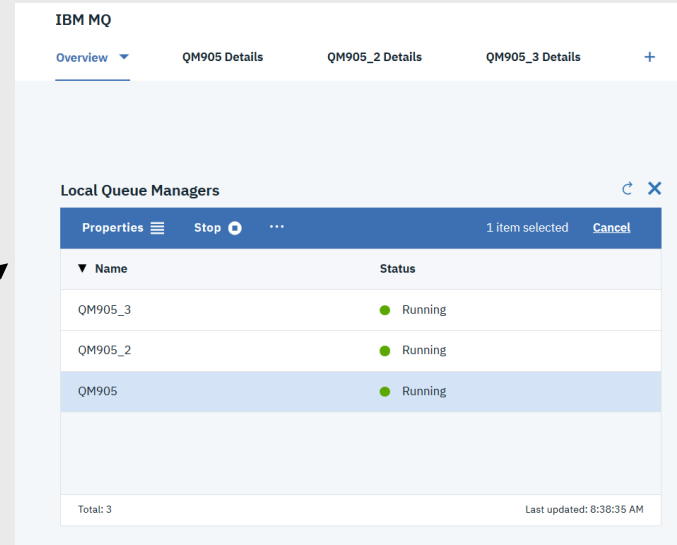
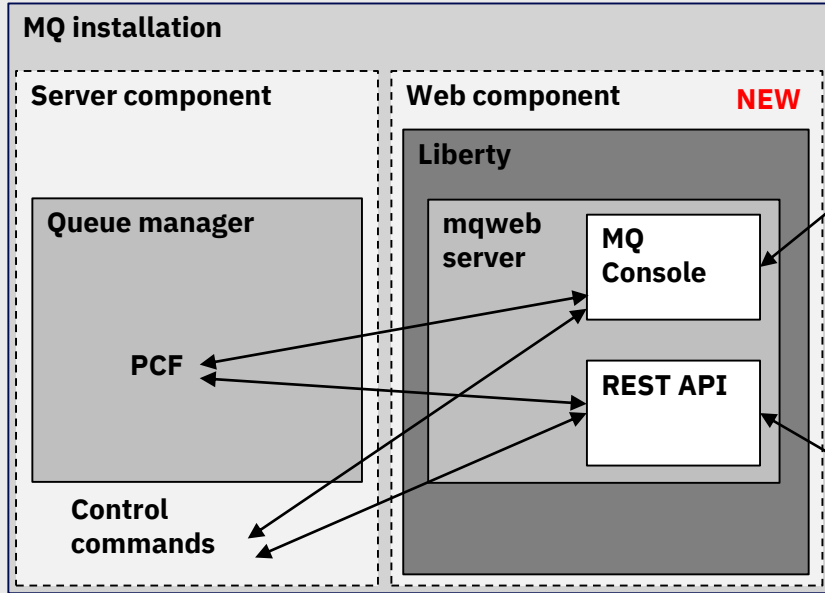
The mqweb server

Web component

- A new optional install component
- Contains the MQ Console, MQ REST APIs plus prereqs
 - WebSphere Liberty Profile which runs the mqweb server
- New USS FMID on z/OS
- JMS9016



Perhaps a picture would help?



HTTP

HTTP

```
C:\>curl -k https://localhost:9443/ibmmq/rest/v1/admin/qmgr/QM905 -u mqadmin:mqadmin
{"qmgr": [{"name": "QM905", "state": "running"}]}
```

The mqweb server

- The MQ Console and REST API are applications that run in a WebSphere Liberty Profile (WLP) server called mqweb
 - WLP is provided as part of MQ install
 - mqweb server definition provided out of the box when installing the web component
- Once installed
 - MQ Console is enabled
 - REST API is enabled
 - Locked down

CWWKE0001I: The server mqweb has been launched.

CWWKG0028A: Processing included configuration resource: C:\Program Files\IBM\Latest902\web\mq\etc\mqweb.xml

A CWWKG0028A: Processing included configuration resource: C:\Program Files (x86)\IBM\WebSphere MQ\web\installations\Latest902\servers\mqweb\mqwebuser.xml

CWWKE0002I: The kernel started after 2.493 seconds

CWWKF0007I: Feature update started.

CWWKO0219I: TCP Channel defaultHttpEndpoint-ssl has been started and is now listening for requests on host 127.0.0.1 (IPv4: 127.0.0.1) port 9443.

CWWKZ0018I: Starting application com.ibm.mq.rest.

CWWKZ0018I: Starting application com.ibm.mq.console.

SRVE0169I: Loading Web Module: com.ibm.mq.rest.v1.

SRVE0250I: Web Module com.ibm.mq.rest.v1 has been bound to default_host.

CWWKT0016I: Web application available (default_host): <https://localhost:9443/ibmmq/rest/v1/>

CWWKZ0001I: Application com.ibm.mq.rest started in 0.518 seconds.

SRVE0169I: Loading Web Module: mqconsole.

SRVE0250I: Web Module mqconsole has been bound to default_host.

CWWKT0016I: Web application available (default_host): <https://localhost:9443/ibmmq/console/>

SRVE0169I: Loading Web Module: com.ibm.mq.consoleinternal.

SRVE0250I: Web Module com.ibm.mq.consoleinternal has been bound to default_host.

CWWKT0016I: Web application available (default_host): <https://localhost:9443/ibmmq/console/internal/>

CWWKZ0001I: Application com.ibm.mq.console started in 0.525 seconds.

CWWKF0012I: The server installed the following features: [concurrent-1.0, jsp-2.2, servlet-3.1, ssl-1.0, jndi-1.0, basicAuthenticationMQ-1.0, websocket-1.0, json-1.0, localConnector-1.0, jaxrs-1.1].

CWWKF0008I: Feature update completed in 2.095 seconds.

CWWKF0011I: The server mqweb is ready to run a smarter planet.

Configuring the mqweb server

- Currently done by editing xml (standard WLP approach)
- Although we have provided commands for setting and displaying certain properties
 - setmqweb and dspmqweb
- File called mqwebuser.xml provided in MQ data directory
- This is the only part of the WLP xml configuration that we support customers editing:

```
<!--  
  Sample mqwebuser.xml file, included by server.xml, to contain user  
  configuration for the mqweb server.  
-->  
  
<server>  
  <featureManager>  
    <feature>appSecurity-2.0</feature>  
  </featureManager>  
  
<!--  
  Default MQ security configuration allows HTTPS TLS v1.2 ONLY and no user access,  
  refer to the IBM Knowledge Center section on "IBM MQ Console and REST API security"  
  for details of how to configure security.  
-->  
  
  <sslDefault sslRef="mqDefaultSSLConfig"/>  
  <basicRegistry id="basic" realm="defaultRealm"/>  
  
</server>
```

Security enabled by default

No users defined

Managing the mqweb server

- Distributed: new control commands
 - strmqweb, endmqweb
- z/OS: Sample JCL – CSQ4WEBS – provided
 - Sets all necessary variables up and then starts up mqweb server
- Also dspmqweb command on all platforms for basic monitoring

```
C:\Program Files\IBM\Latest902\bin>strmqweb.bat
Starting server mqweb.
Server mqweb started.

C:\Program Files\IBM\Latest902\bin>dspmqweb.bat
Server mqweb is running.
URLs:
https://localhost:9443/ibmmq/console/
https://localhost:9443/ibmmq/rest/v1/
```

```
EDIT          USER.PROCLIB(MQWEBMWL) - 01.03
Command ==>
000058 //*****
000059 //*
000060 //          PROC
000061 //*
000062 // SET INSTDIR='/u/mleming/v902betamqm/web'
000063 // SET USERDIR='/u/mleming/mq902web'
000064 //*
000065 //STEP1 EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT,
000066 // PARM='PGM &INSTDIR./lib/native/zos/s390x/bbgzsrv mqweb'
000067 //WLPUDIR DD PATH='&USERDIR.'
000068 //STEPLIB DD DSN=ANTZ.MQ.V900.DFCT.OUT.SCSQANLE,DISP=SHR
000069 // DD DSN=ANTZ.MQ.V900.DFCT.OUT.SCSQAUTH,DISP=SHR
000070 //STDOUT DD SYSOUT=*
000071 //STDERR DD SYSOUT=*
000072 //STDIN DD DUMMY
000073 //STDENV DD *
000074 JAVA_HOME=/java/java80_bit64_sr3_fp20/J8.0_64
000075 PATH=/u/mleming/v902betamqm/web/bin:/usr/sbin
000076 LIBPATH=/u/mleming/v902betamqm/java/lib
000077 //*
```

The MQ Administrative REST API

What is REST?

- REpresentational State Transfer
 - Term first coined by Roy Fielding in his PhD thesis
 - An architectural style
 - Based off his earlier work defining the HTTP and other web based specifications
- HTTP is an example of a RESTful architecture
- HTTP defines resources (URL/URIs) and the operations (HTTP verbs) which can use them
 - Originally used for serving web-pages
 - Works really well for APIs too
- Generally lightweight and relatively simple to use, much simpler than SOAP web-services
 - Have become incredibly common in recent years
- However there are lots of interpretations of what it means to be RESTful
 - MQ has taken the approach of following best-practice, and adherence to the various w3c standards when defining its REST API

MQ Admin REST API

- An administrative API for managing MQ via REST
 - Messaging API added in 9.0.4
- Is much more intuitive to use than PCF and makes it easier to create MQ tooling, e.g. a self-service web-browser based MQ portal using JavaScript
 - No need for an MQ client!
 - Callable from any language which can invoke an HTTPS endpoint
 - Most languages now have built in, or easily added, support for REST
- Payload format is JSON (JavaScript Object Notation)
 - Human readable, not a binary format

Curly bracket denotes JSON object

Square bracket denotes JSON array

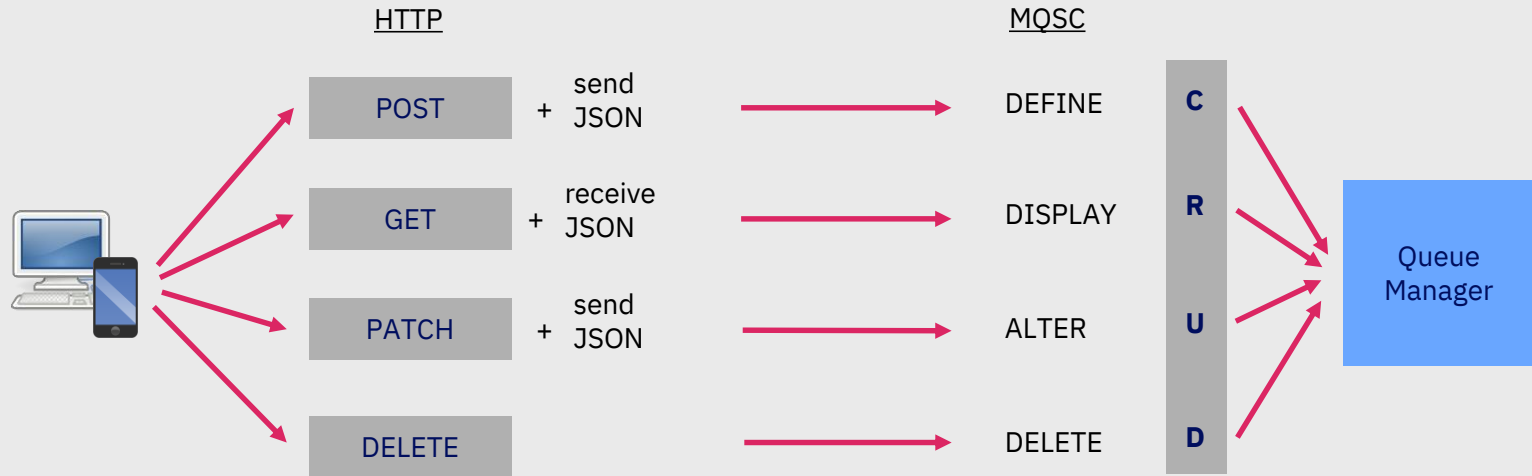
A nested unnamed object, in an array

```
{  
  "qmgr": [  
    {  
      "name": "AQM1",  
      "state": "running"  
    }  
  ]  
}
```

Name, value pair. Where value is of type string

MQ Admin REST API

- Based on underlying MQ capabilities such as PCF and control commands, but adjusted to adhere to RESTful practices
- URL represent target object for command



Evolution of the MQ REST APIs

- Iteratively developed in CD releases

- 9.0.1:
 - REST API for administration introduced
 - Contains ability to list queue managers (dspmq) and their installation (dspmqver)
 - Not integrated into mqweb server/MQ security so disabled by default
- 9.0.2:
 - Integrated into mqweb server and MQ security, enabled by default
 - Contains CRUD for queues and the ability to display queue status
 - Supported on MQ Appliance
- 9.0.3:
 - Support for subset of DIS QMSTATUS on all platforms including z/OS
- 9.0.4
 - REST API for messaging introduced
 - Administration API further enhanced
 - Ability to run MQSC commands
 - Ability to display channels and subscriptions
 - REST API administration gateway introduced
- 9.0.5
 - REST API for MFT administration introduced
 - Administration gateway support expanded
- 9.1
 - Fully supported on all platforms in LTS

Some examples...

GET /ibmmq/rest/v1/admin/qmgr (dspmq)

- Ability to list queue managers associated with installation
- Example below uses curl to list all queue managers
- -k flag tells it to ignore the fact that a self-signed certificate is being used on the mqweb server, you don't want to be doing this in production!

```
C:\>curl -k https://localhost:9443/ibmmq/rest/v1/admin/qmgr -u mqadmin:mqadmin
{"qmgr": [
  {
    "name": "QM905",
    "state": "running"
  },
  {
    "name": "QM905_2",
    "state": "running"
  },
  {
    "name": "QM905_3",
    "state": "running"
  }
]}
```

GET /ibmmq/rest/v1/admin/qmgr (dspmq)

- Can get information on just a specific queue manager
 - GET /ibmmq/rest/v1/admin/qmgr/{qmgrName}
- Can request additional attributes too, or just a sub-set
 - GET /ibmmq/rest/v1/admin/qmgr?attributes=*

```
C:\>curl -k https://localhost:9443/ibmmq/rest/v1/admin/qmgr?attributes=* -u mqadmin:mqadmin
{"qmgr": [
  {
    "extended": {
      "installationName": "MQ905",
      "isDefaultQmgr": false,
      "permitStandby": "notPermitted"
    },
    "name": "QM905",
    "state": "running"
  },
  {
    "extended": {
      "installationName": "MQ905",
      "isDefaultQmgr": false,
      "permitStandby": "notPermitted"
    },
    "name": "QM905_2",
    "state": "running"
  }
],
```

GET /ibmmq/rest/v1/admin/installation (dspmqver)

- Basic display

```
C:\>curl -k https://localhost:9443/ibmmq/rest/v1/admin/installation -u mqadmin:mqadmin
{"installation": [{
  "name": "MQ905",
  "platform": "windows",
  "version": "9.0.5.0"
}]}
```

- All attributes

```
C:\>curl -k https://localhost:9443/ibmmq/rest/v1/admin/installation?attributes=* -u mqadmin:mqadmin
{"installation": [{
  "extended": {
    "dataPath": "C:\\ProgramData\\IBM\\MQ",
    "description": "",
    "hostName": "localhost",
    "installationPath": "C:\\Program Files\\IBM\\MQ905",
    "level": "p905-L180305.1",
    "maximumCommandLevel": 905,
    "operatingSystem": "Windows 7 Professional x64 Edition, Build 7601: SP1",
    "primary": false
  },
  "name": "MQ905",
  "platform": "windows",
  "version": "9.0.5.0"
}]}
```

Queues...

DEFINE Q*

– POST to /ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue

```
curl -k -X POST -H "Content-Type: application/json" -d "{\"name\": \"Q1\"}"  
https://localhost:9443/ibmmq/rest/v1/admin/qmgr/MQ905/queue
```

Sending JSON payload

Queue manager name

Queue definition, very simple in this case

DISPLAY Q*

– GET to /ibmmq/rest/v1/qmgr/admin/{qmgrName}/queue/{queueName}

```
C:\>curl -k https://localhost:9443/ibmmq/rest/v1/admin/qmgr/MQ905/queue?name=Q* -u mqadmin:mqadmin  
{"queue": [{"  
  "name": "Q1",  
  "type": "local"  
}]}
```

Queues...

ALTER Q*

- PATCH to `/ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue/{queueName}`
- E.g: the following will PUT inhibit Q.LOCAL1

```
curl -k -X PATCH -H "Content-Type: application/json" -d "{\"general\":{\"inhibitPut\": true}}"
https://localhost:9443/ibmmq/rest/v1/admin/qmgr/QM905/queue/Q.LOCAL1
```

DELETE Q*

- DELETE to `/ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue/{queueName}`

```
C:\>curl -k -X DELETE https://localhost:9443/ibmmq/rest/v1/admin/qmgr/QM905/queue/AQ -u mqadmin:mqadmin -H "ibm-mq-rest-csrf-token: value"
C:\>curl -k -X DELETE https://localhost:9443/ibmmq/rest/v1/admin/qmgr/QM905/queue/AQ -u mqadmin:mqadmin -H "ibm-mq-rest-csrf-token: value"
{"error": [{"
  "action": "Resubmit the request with the name of an existing queue, or with no queue name to retrieve a list of queues.",
  "completionCode": 2,
  "explanation": "The MQ REST API was invoked specifying a queue name which cannot be located.",
  "message": "MQWB0037E: Could not find the queue 'AQ' - the queue manager reason code is 2085 : 'MQRC_UNKNOWN_OBJECT_NAME'.",
  "msgId": "MQWB0037E",
  "reasonCode": 2085,
  "type": "rest"
}]}
```

Queues...

Also possible to issue DISPLAY QSTATUS

- GET to
/ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue/{queueName}?status=* &applicationHandle=*
- So you can get both the queue definition and its status at the same time!

```
C:\>curl -k https://localhost:9443/ibmmq/rest/v1/admin/qmgr/QM905/queue/Q1?status=* -u mqadmin:mqadmin
{"queue": [{"name": "Q1", "status": {"currentDepth": 0, "lastGet": "", "lastPut": "", "mediaRecoveryLogExtent": "", "monitoringRate": "off", "oldestMessageAge": -1, "onQueueTime": {"longSamplePeriod": -1, "shortSamplePeriod": -1}, "openInputCount": 0, "openOutputCount": 0, "uncommittedMessages": 0}, "type": "local"}]}
```


MQSC for REST

Tailored RESTful support for individual MQ objects and actions are in the works...

However, to speed up full MQ admin support over REST we will be adding the ability to submit arbitrary MQSC commands over REST

- ✓ Gives complete MQSC coverage quickly
- ✓ Simple to convert existing scripts
- ✗ Does not benefit from improved usability

HTTPS POST:

<https://host:port/ibmmq/v1/admin/action/qmgr/QMGR1/mqsc>

```
{
  "type": "runCommand",
  "parameters": {
    "command": "STOP CHANNEL(CHANNEL.TEST)"
  }
}
```

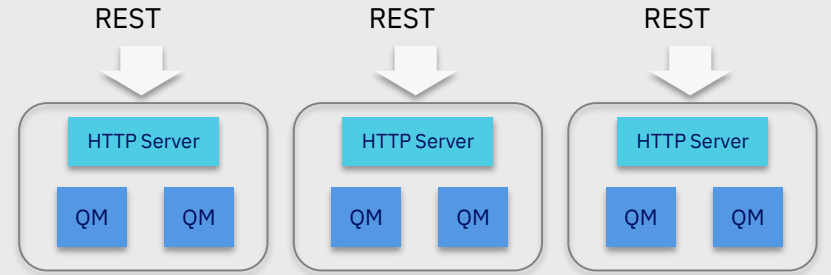
```
{
  "commandResponse": [{
    "completionCode": 0,
    "reasonCode": 0,
    "text": ["AMQ8019: Stop IBM MQ channel accepted."]
  }],
  "overallCompletionCode" : 0,
  "overallReasonCode" : 0
}
```

Stopping a channel

Enabling your whole estate for REST administration

Option 1

Administer each MQ installation separately

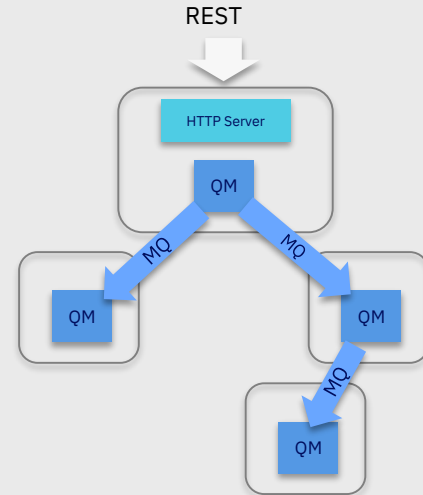


Option 2

Manage a network of systems through gateway entry points

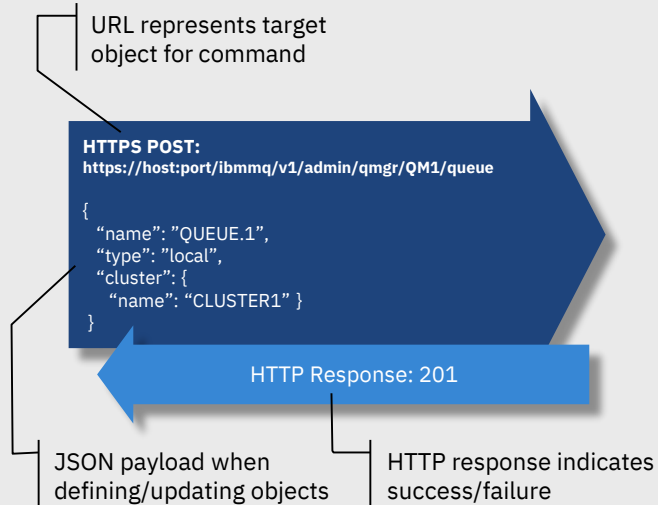
Not every queue manager will need to expose HTTPS endpoints

Pre-version 9.1 queue managers are able to be administered through 9.1 gateways



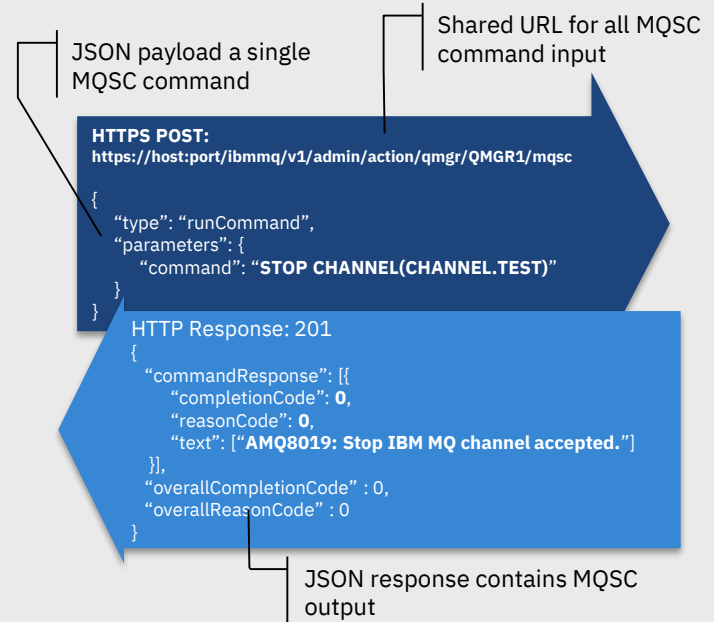
Two approaches

Per object REST



Native JSON based REST calls

MQSC over REST



Direct MQSC command input over REST
MQSC output over REST, minimal parsing

Two approaches

Per object REST

- Natural REST APIs
 - Restructured definitions to aid understanding
 - Further definition validation
 - Not a straight swap for existing users
 - Incomplete coverage of MQ administration
- URL represents target object for command
- JSON payload when defining/updating objects
- HTTP response indicates success/failure
- ```
https://host:port/ibmmq/v1/admin/qmgr/QMGR1/queue
{"type": "local",
 "cluster": {
 "name": "LOCALHEEP1",
 "description": "LOCALHEEP1",
 "httpResponse": 201
 }
}
```

Native JSON based REST calls

...

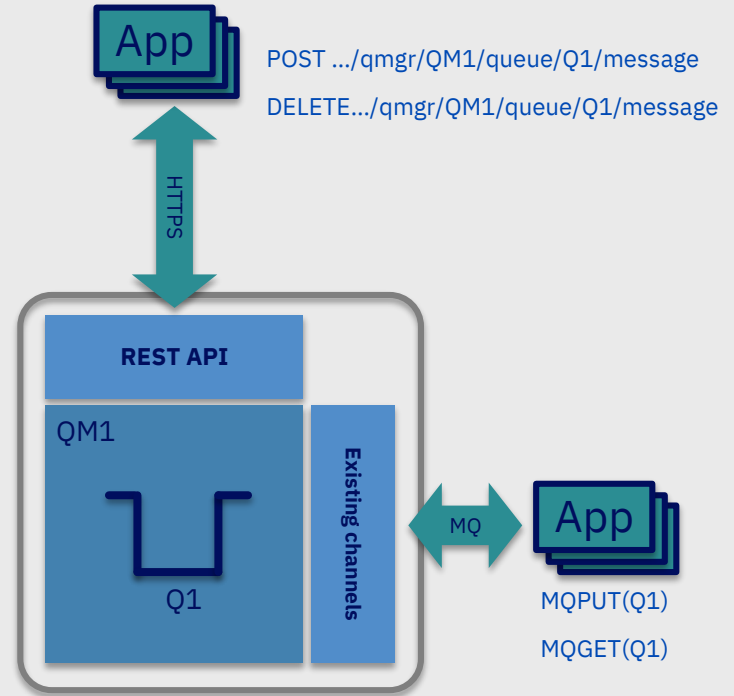
## MQSC over REST

- Simple mapping from existing scripts
  - Complete coverage of MQSC capabilities
  - Not pure REST
  - Just as *simple* as existing runmqsc for input and parsing of output
- JSON payload a single MQSC command
- Shared URL for all MQSC command input
- ```
HTTPS POST:  
https://host:port/ibmmq/v1/admin/action/qmgr/QMGR1/mqsc  
{"parameters": {  
  "command": "STOP CHANNEL(CHANNEL.TEST)"  
},  
 "overallCompletionCode": 0,  
 "overallReasonCode": 0,  
 "text": ["AMQ8019: Stop IBM MQ channel accepted."]  
}
```
- JSON response contains MQSC output

Direct MQSC command input over REST
MQSC output over REST, minimal parsing

REST Messaging

- The new HTTP server support in MQ 9.0.x provides the platform for a properly integrated REST API solution
- Allowing applications to put and get messages from a queue without installing any MQ software locally
- Ideal for environments with native REST support, such as common JavaScript libraries including NodeJS, and AngularJS
- Can only be used for point to point messaging
- For full functionality and resiliency an MQ client should still be used



API Discovery

API discovery

Want to find out what is available in the MQ REST API, and don't want to read the KC?

Then try out API discovery!

Function in WLP that describes the MQ REST API using Swagger

Makes it easier to see what is there, and try it out

API Discovery : APIs available from the API Discovery feature		Show/Hide	List Operations	Expand Operations
channel		Show/Hide	List Operations	Expand Operations
installation		Show/Hide	List Operations	Expand Operations
login		Show/Hide	List Operations	Expand Operations
messaging		Show/Hide	List Operations	Expand Operations
mft : agent		Show/Hide	List Operations	Expand Operations
mft : transfer		Show/Hide	List Operations	Expand Operations
qmgr		Show/Hide	List Operations	Expand Operations
qmgr : action		Show/Hide	List Operations	Expand Operations
queue		Show/Hide	List Operations	Expand Operations
GET	/ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue			Retrieves details of all queues.
POST	/ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue			Creates a queue.
DELETE	/ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue/{qName}			Deletes a queue.
GET	/ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue/{qName}			Retrieves details of a specific queue.
PATCH	/ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue/{qName}			Modifies a queue.

API discovery

GET /ibmmq/rest/v1/admin/qmgr/{qmgrName}/queue

Retrieves details of all queues.

Implementation Notes

Retrieves details of all queues defined in the named queue manager, optionally specifying which attributes of the queues are to be retrieved.

Response Class (Status 200)

A JSONArray containing a JSONObject describing the queue.

Model | Example Value

```
{
  "cluster": {
    "namelist": "string",
    "qmgrId": "string",
    "name": "string",
    "qmgrName": "string",
    "queueType": "string",
    "workloadQueueUse": "string",
    "workloadPriority": 0,
    "transmissionQueueForChannelName": "string",
```

Response Content Type application/json; charset=utf-8

Parameters

Parameter	Value	Description	Parameter Type	Data Type
qmgrName	<input type="text" value="(required)"/>	Name of the queue manager containing the queue of interest.	path	string

Security

REST API security

Role based access control. Need to be a member of at least one role

- MQWebAdmin
- MQWebAdminRO
- MQWebUser
- MFTWebAdmin
- MFTWebAdminRO

User and groups defined in a registry

- Basic
- LDAP
- SAF (on z/OS)
- OS (on distributed)

REST is locked down by default, need to do some configuring

- Samples provided to make this simpler

```
<!--  
Roles for the MQ REST API  
-->  
<enterpriseApplication id="com.ibm.mq.rest">  
  <application-bnd>  
    <security-role name="MQWebAdmin">  
      <group name="MQWebAdminGroup" realm="defaultRealm"/>  
    </security-role>  
    <security-role name="MQWebAdminRO">  
      <user name="mqreader" realm="defaultRealm"/>  
    </security-role>  
    <security-role name="MQWebUser">  
      <special-subject type="ALL_AUTHENTICATED_USERS"/>  
    </security-role>  
    <security-role name="MFTWebAdmin">  
      <user name="mftadmin" realm="defaultRealm"/>  
    </security-role>  
    <security-role name="MFTWebAdminRO">  
      <user name="mftreader" realm="defaultRealm"/>  
    </security-role>  
  </application-bnd>  
</enterpriseApplication>
```

```
<!-- Sample Basic Registry -->  
<basicRegistry id="basic" realm="defaultRealm">  
  <!-- This sample defines two users with unencoded passwords -->  
  <!-- and a group, these are used by the role mappings above -->  
  <user name="mqadmin" password="mqadmin"/>  
  <user name="mqreader" password="mqreader"/>  
  <group name="MQWebUI">  
    <member name="mqadmin"/>  
  </group>  
</basicRegistry>
```

```
<!-- Example LDAP Registry -->  
<ldapRegistry id="ldap"  
  realm="MyOrganizationRealm"  
  host="sso.example.com"  
  port="389"  
  ignoreCase="true"  
  baseDN="o=example.com"  
  certificateMapMode="EXACT_DN"  
  ldapType="IBM Tivoli Directory Server"  
  idsFilters="ibm_dir_server">  
</ldapRegistry>
```

REST API authentication

Token based

- User logs in once with user id and password and then gets a cookie which is used for subsequent requests

```
curl -k -X POST -H "Content-Type: application/json"
```

```
-d '{"username":"mqadmin\","password":"mqadmin\"}'
```

```
https://localhost:9443/ibmmq/rest/v1/login -c c:\temp\cookiejar.txt
```

User id and password provided as JSON payload

Cookie stored for use on next request

- DELETE to the login URL logs out

Or HTTP basic authentication

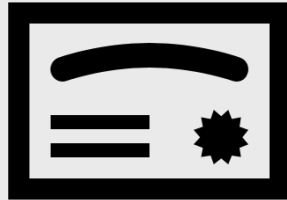
- User id and password provided as an encoded header, must be set for each request

```
C:\>curl -k https://localhost:9443/ibmmq/rest/v1/admin/installation -u mqadmin:mqadmin  
{  
  "installation": [  
    {  
      "name": "MQ905",  
      "platform": "windows",  
      "version": "9.0.5.0"  
    }  
  ]  
}
```

REST API authentication

Or use a client certificate

- Must be provided with each call to the REST API
- Distinguished name from certificate is mapped to user in configured user registry



The MQ Console

MQ Console

- Browser based interface for administering and managing MQ
 - No client side install needed
 - Originally available in MQ Appliance only
- As of 9.0.1 a common capability across appliance and software MQ
 - Re-engineered on AngularJS so different implementation than on 8.0.0.* appliance
 - Functional parity with MQ Console in 8.0.0.* appliance
- Some capabilities not available on z/OS
 - Can't create/delete/start/stop queue managers, etc.
- Can only interact with queue managers running in the same installation
 - On z/OS all queue managers at the same VRM level

MQ Console – log in

Point your web-browser at the MQ Console and log in

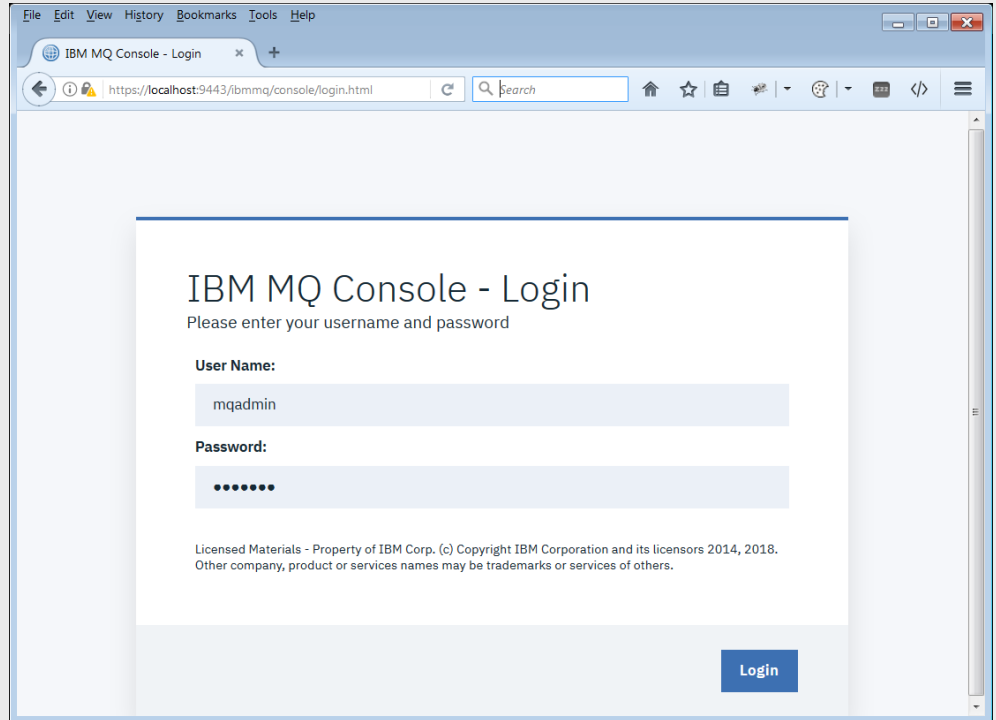
- With a user id and password
- With a client certificate

Log in credentials validated via user registry configured in the mqweb server

- Like the REST API

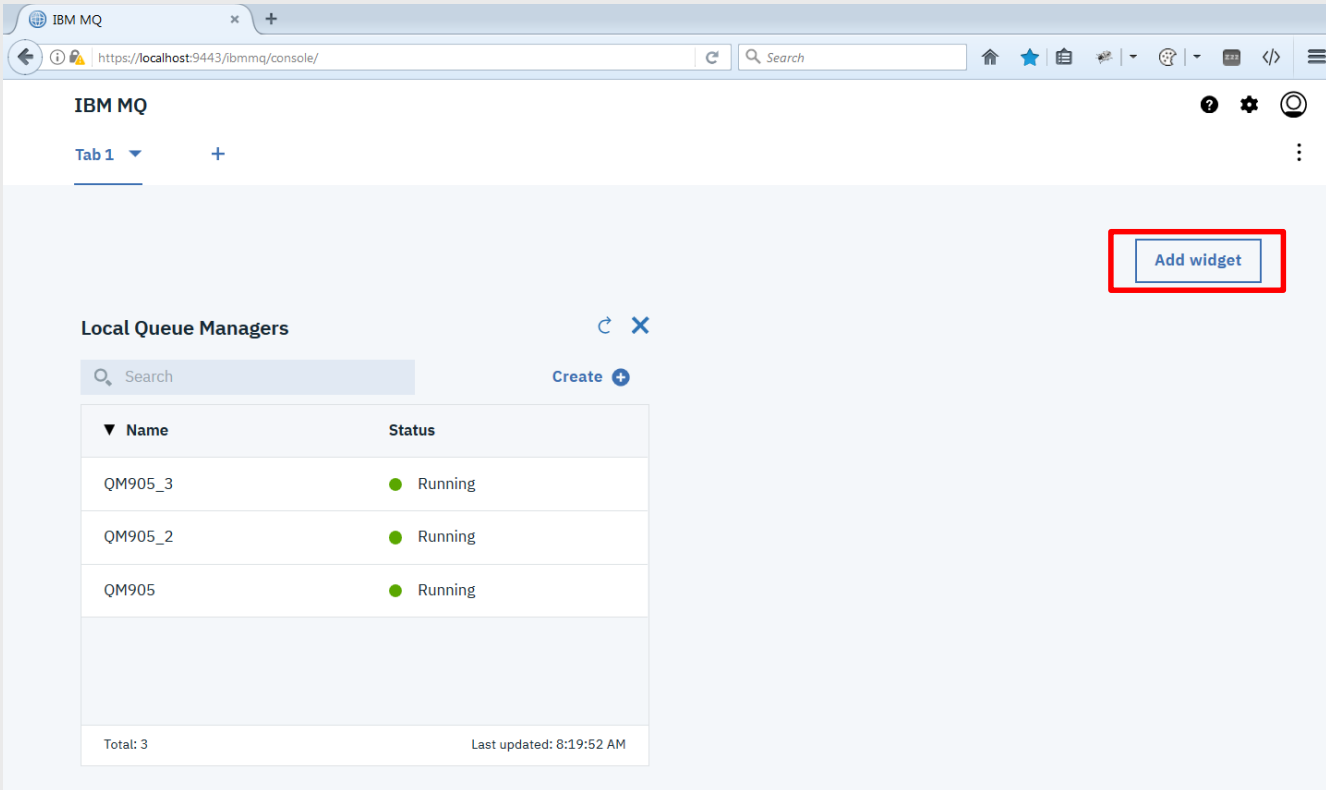
Access determined by role

- Same role names as REST API
- But in a different name space so REST users don't need to have same access as MQ Console users



MQ Console – add widgets

Console dashboard consists of a number of widgets, each widget shows information for a particular set of MQ objects: queue managers, queues, etc.



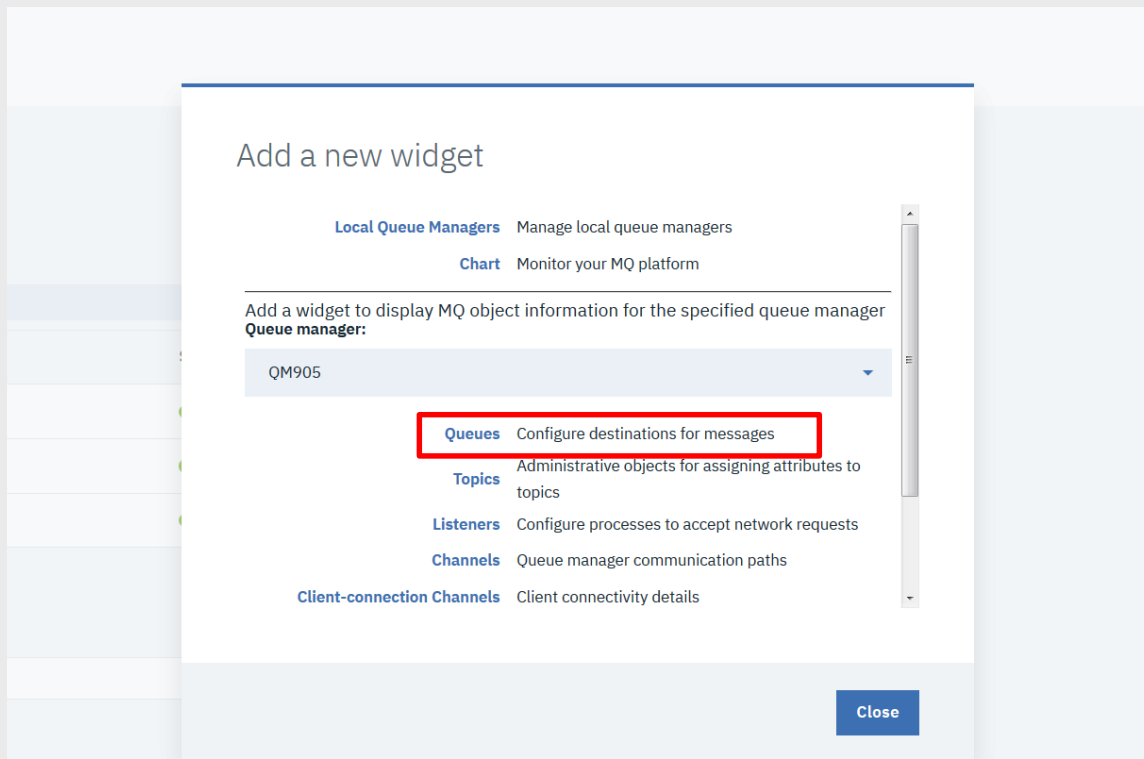
The screenshot shows the IBM MQ Console interface in a web browser. The browser address bar displays `https://localhost:9443/ibmmq/console/`. The page title is "IBM MQ". Below the title, there is a tab labeled "Tab 1" and a "+" icon to add more tabs. In the top right corner, there are icons for help, settings, and refresh. A red rectangular box highlights a blue "Add widget" button in the upper right area of the dashboard. Below this, the "Local Queue Managers" section is visible, featuring a search bar, a "Create +" button, and a table with the following data:

Name	Status
QM905_3	● Running
QM905_2	● Running
QM905	● Running

At the bottom of the table, it shows "Total: 3" and "Last updated: 8:19:52 AM".

MQ Console – add widgets

Console dashboard consists of a number of widgets, each widget shows information for a particular set of MQ objects: queue managers, queues, etc.



MQ Console – add widgets

Console dashboard consists of a number of widgets, each widget shows information for a particular set of MQ objects: queue managers, queues, etc.

The screenshot shows the IBM MQ Console interface. At the top, there's a browser tab for 'IBM MQ' and a search bar. Below the header, there's a 'Tab 1' dropdown and an 'Add widget' button. The main content area contains two widgets:

- Local Queue Managers:** A table with columns 'Name' and 'Status'. It lists three queue managers: QM905_3, QM905_2, and QM905, all with a 'Running' status. A 'Total: 3' and 'Last updated: 8:22:10 AM' are shown at the bottom.
- Queues on QM905:** A table with columns 'Name', 'Queue type', and 'Queue depth'. It lists six queues: AQ, MAXINSTQ, NOTIFICATIONQ, NOfificationQ, and Q1. All are 'Local' type. Queue depths are 2, 30, 0, 0, and 0 respectively. A 'Total: 7' and 'Last updated: 8:21:52 AM' are shown at the bottom.

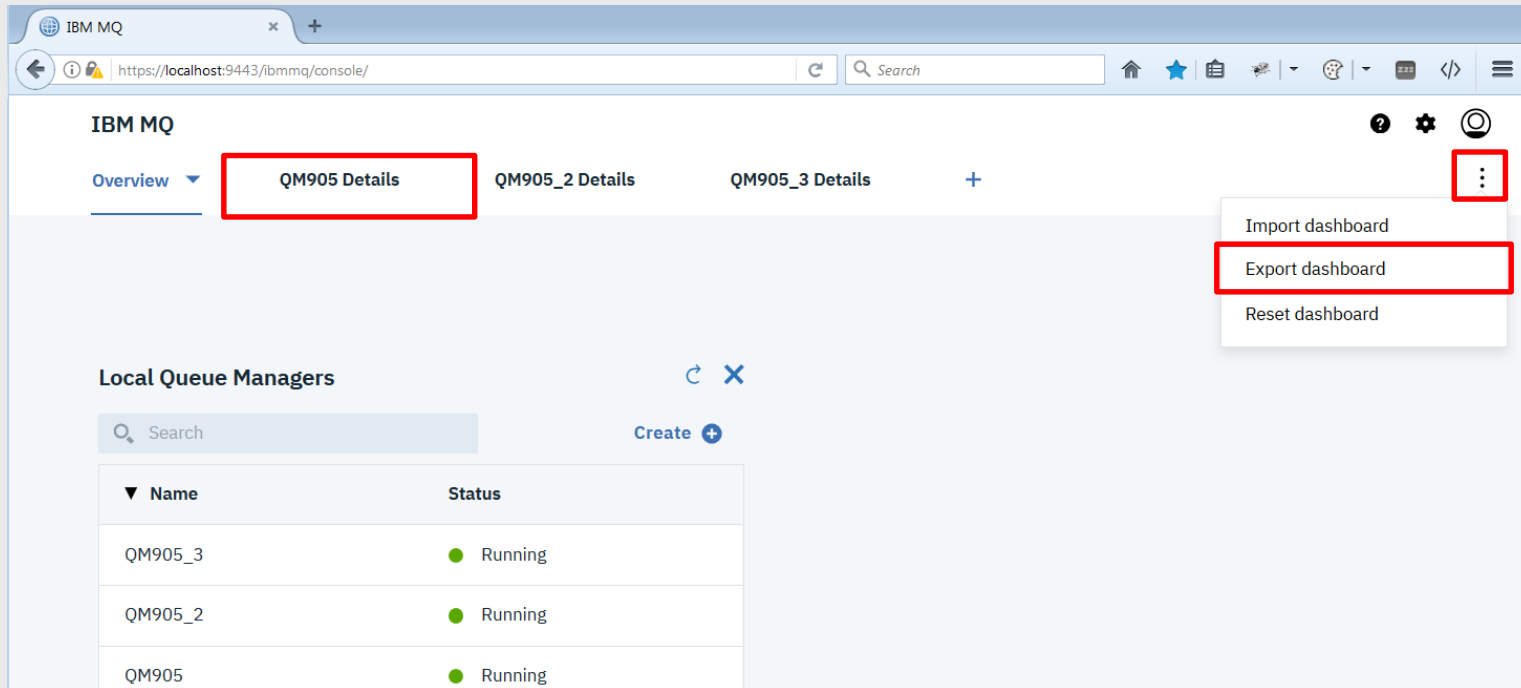
The 'Queues on QM905' widget is highlighted with a red border.

MQ Console – layout

Can use multiple tabs to help manage content

Each user can lay out their dashboard according to **their** needs

Can export dashboard to share layout with others



The screenshot shows the IBM MQ Console interface in a web browser. The browser address bar displays `https://localhost:9443/ibmmq/console/`. The main header area includes the text "IBM MQ" and a navigation menu with tabs: "Overview", "QM905 Details", "QM905_2 Details", and "QM905_3 Details". The "QM905 Details" tab is highlighted with a red box. To the right of the tabs is a vertical ellipsis menu icon, also highlighted with a red box. A dropdown menu is open, showing three options: "Import dashboard", "Export dashboard", and "Reset dashboard". The "Export dashboard" option is highlighted with a red box. Below the navigation menu is a section titled "Local Queue Managers" with a search bar and a "Create +" button. A table lists the queue managers and their status:

Name	Status
QM905_3	Running
QM905_2	Running
QM905	Running

MQ Console – manage

Monitor your MQ queue managers using charts generated from statistics information published to system topics

– added in 9.0.0 on distributed platforms

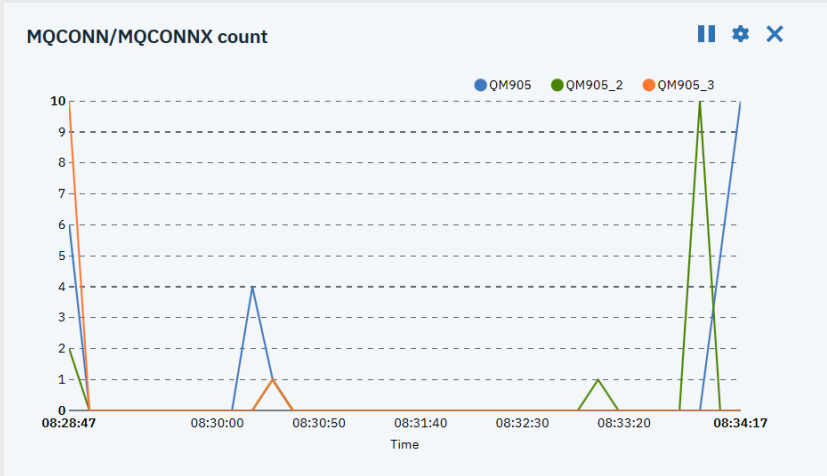
Display and alter objects using the properties editor

Browse and send messages

Properties for 'QM905_3'

- General
- Extended
- Cluster
- Repository
- Communication
- Events
- SSL
- Statistics
- Online monitoring
- Statistics monitoring
- Accounting monitoring

Default transmission queue:	Channel auto definition:
	Disabled
Channel auto definition exit:	IP address version:
	IPv4
Activity recording:	Trace-route recording:
Message	Message
CHLAUTH records:	REVDNS lookup:
Enabled	Enabled



Subscribe to Topic

Topic string: *

/test/topic Refresh

Search

Date/Time	Message Body
May 18, 2018 8:37:11 AM	Hello world 1
May 18, 2018 8:37:14 AM	Hello world 2
May 18, 2018 8:37:17 AM	Hello world 3

Summary

- Current options, and why we need something else
- The mqweb server
- The MQ administrative REST API
- Examples
- API Discovery
- Security
- The MQ Console

We want your feedback!

- Please submit your feedback online at
 - <http://conferences.gse.org.uk/2018/feedback/JJ>
- Paper feedback forms are also available from the Chair person
- This session is JJ



Notices and disclaimers

- © 2018 International Business Machines Corporation.
No part of this document may be reproduced or transmitted in any form without written permission from IBM.
- **U.S. Government Users Restricted Rights — use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.**
- Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. **This document is distributed “as is” without any warranty, either express or implied. In no event, shall IBM be liable for any damage arising from the use of this information, including but not limited to, loss of data, business interruption, loss of profit or loss of opportunity.** IBM products and services are warranted per the terms and conditions of the agreements under which they are provided.
- IBM products are manufactured from new parts or new and used parts.
In some cases, a product may not be new and may have been previously installed. Regardless, our warranty terms apply.”
- **Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.**
- Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those
- customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.
- References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.
- Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.
- It is the customer’s responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer’s business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer follows any law.

Notices and disclaimers continued

- Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products about this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. **IBM expressly disclaims all warranties, expressed or implied, including but not limited to, the implied warranties of merchantability and fitness for a purpose.**
- The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.
- IBM, the IBM logo, ibm.com and [names of other referenced IBM products and services used in the presentation] are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml.
- .