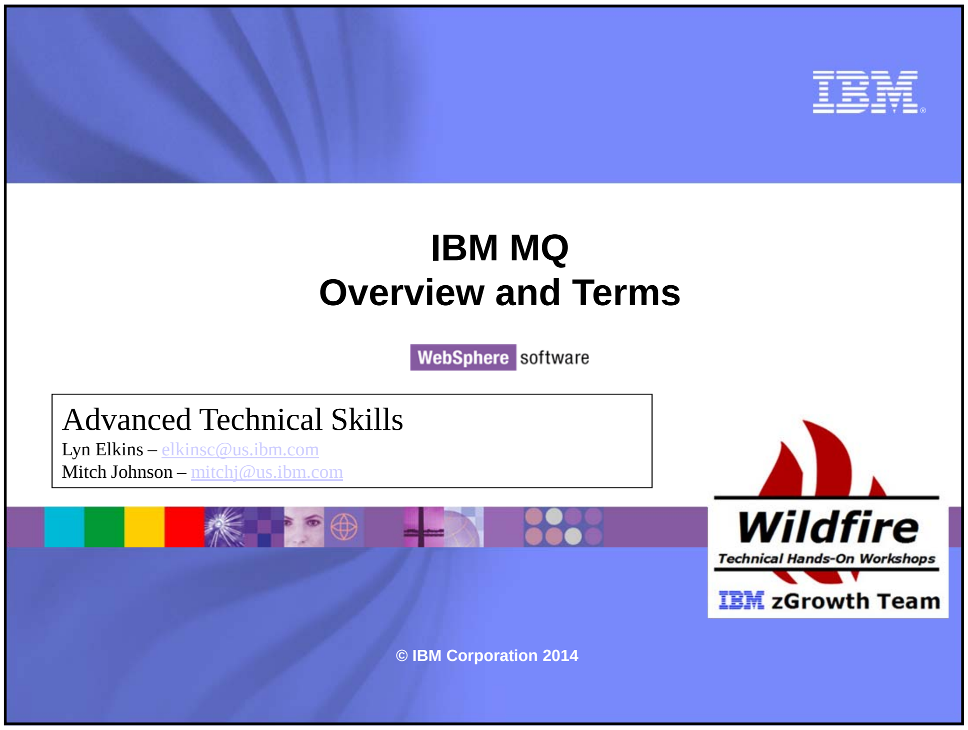
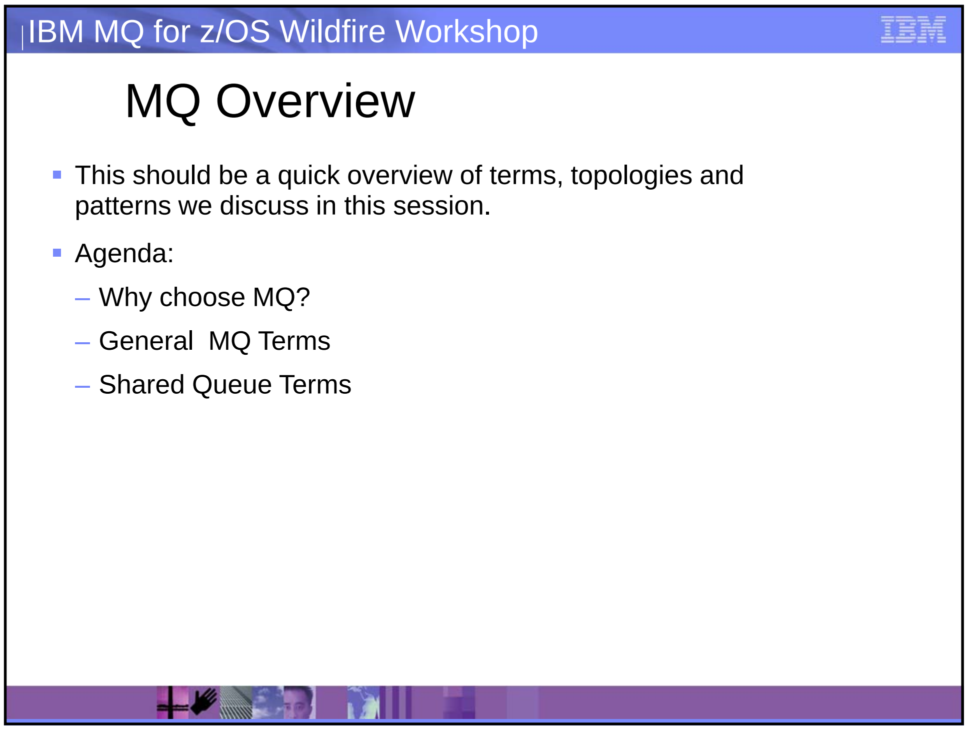


IBM MQ Overview and Terms

WebSphere software

Advanced Technical Skills
Lyn Elkins – elkinsc@us.ibm.com
Mitch Johnson – mitchj@us.ibm.com

Wildfire
Technical Hands-On Workshops
IBM zGrowth Team



MQ Overview

- This should be a quick overview of terms, topologies and patterns we discuss in this session.
- Agenda:
  - Why choose MQ?
  - General MQ Terms
  - Shared Queue Terms

Why choose IBM MQ - Does your infrastructure…

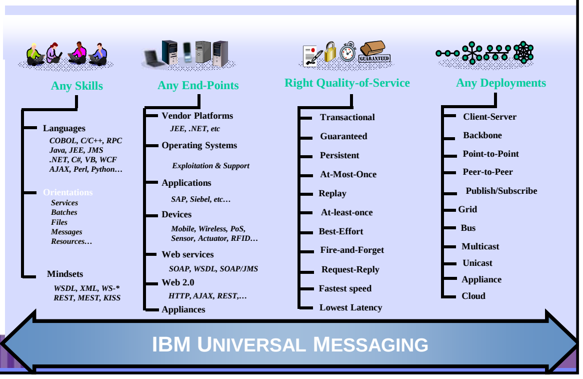
- Use HTTP or Homegrown messaging ***for business-critical events***
- ***Leverage a single backbone*** for all your integration needs
- Allow you to ***reliably*** transport business data
- Ensure your sensitive information and intellectual property ***secure***
- ***Connect decision makers*** to ***real time data*** from the edge
- ***Seamlessly meet*** your fluctuating needs

…Give you the quality of service needed to succeed in this complex and changing environment

Why Choose IBM MQ?

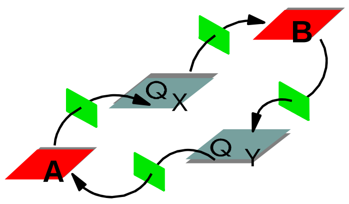
- **Delivers an open, robust, flexible & scalable Universal Messaging Backbone**
- **Connects virtually any commerical IT system**

- **Easy to manage**
- **Shields application developers from networking complexities**
- **Offers a range of Qualities of Service**



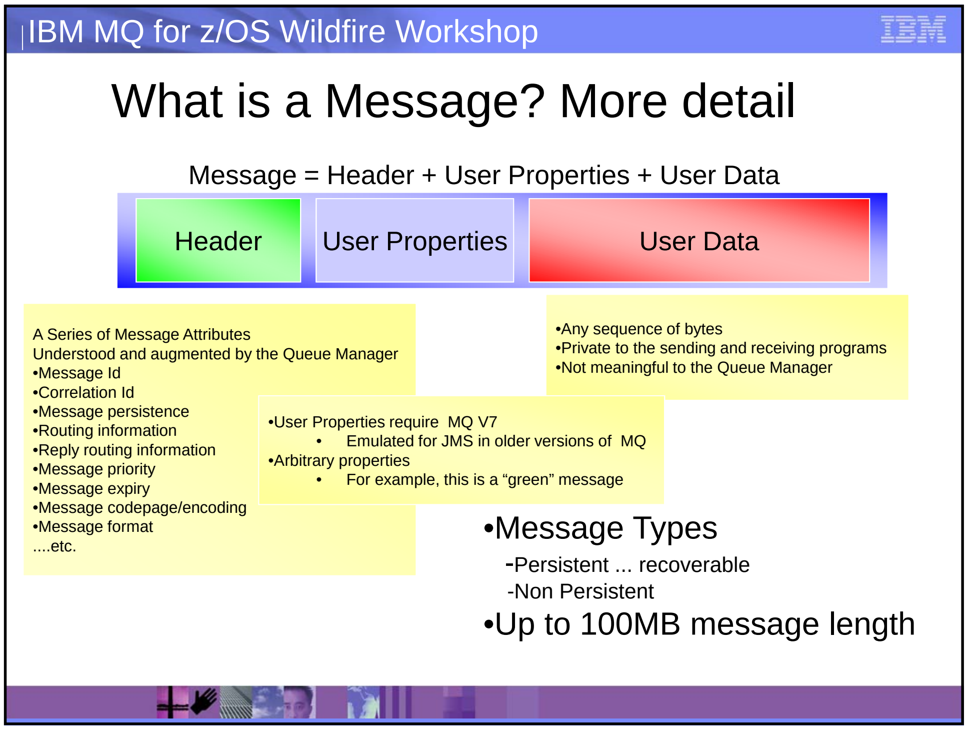
IBM MQ – Terms and Topologies

IBM MQ – Basic Terms



- **Messages** can be created from any source:
  - *Data, Messages, Events, Files, Web service requests / responses*

- Messages are moved asynchronously using **Queues**
- Queues are owned and managed by a **Queue Manager**
- Messages flow between queue managers across **Channels**



IBM MQ for z/OS Wildfire Workshop

What is a Message? More detail

Message = Header + User Properties + User Data

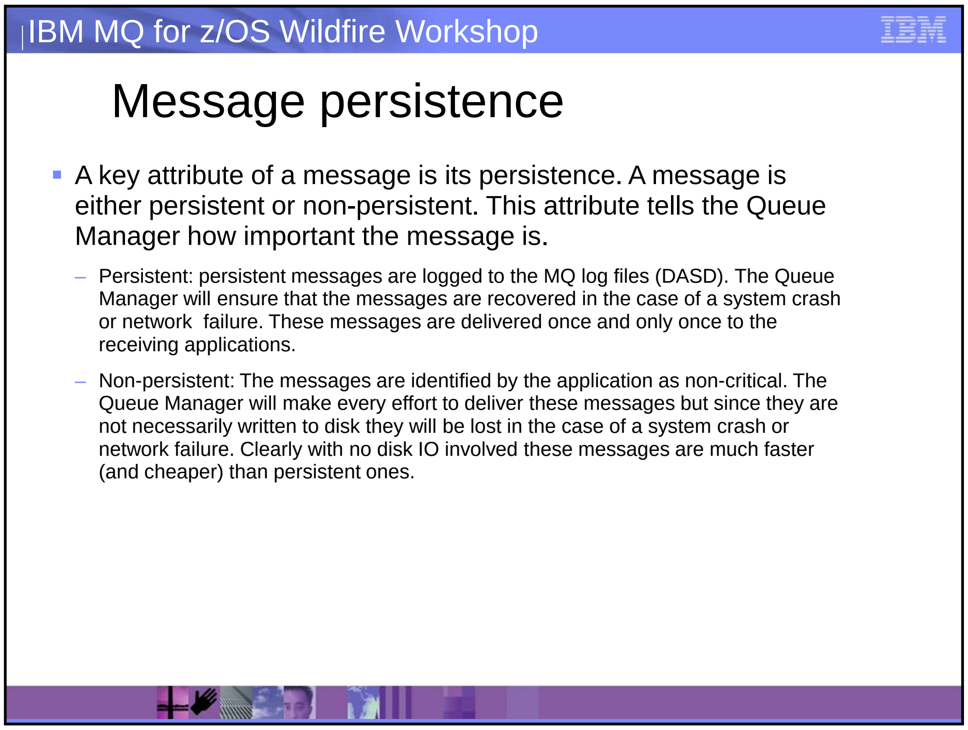
Header | User Properties | User Data

A Series of Message Attributes
Understood and augmented by the Queue Manager
•Message Id
•Correlation Id
•Message persistence
•Routing information
•Reply routing information
•Message priority
•Message expiry
•Message codepage/encoding
•Message format
....etc.

•User Properties require MQ V7
  • Emulated for JMS in older versions of MQ
•Arbitrary properties
  • For example, this is a “green” message

•Any sequence of bytes
•Private to the sending and receiving programs
•Not meaningful to the Queue Manager

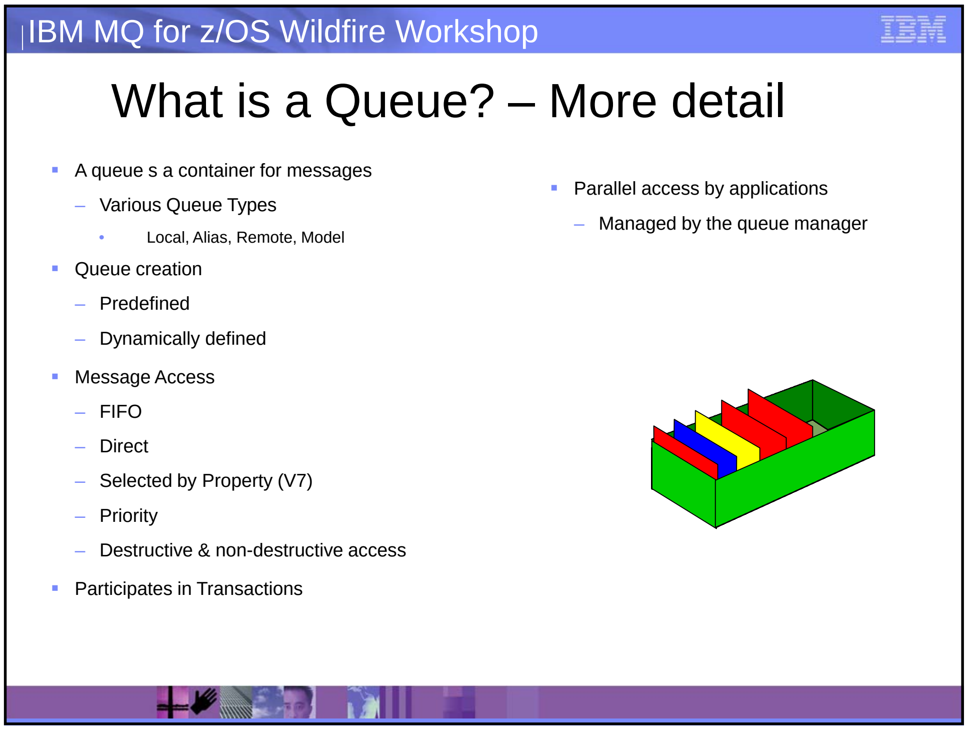
•Message Types
  -Persistent ... recoverable
  -Non Persistent
•Up to 100MB message length



IBM MQ for z/OS Wildfire Workshop

Message persistence

- A key attribute of a message is its persistence. A message is either persistent or non-persistent. This attribute tells the Queue Manager how important the message is.
  – Persistent: persistent messages are logged to the MQ log files (DASD). The Queue Manager will ensure that the messages are recovered in the case of a system crash or network failure. These messages are delivered once and only once to the receiving applications.
  – Non-persistent: The messages are identified by the application as non-critical. The Queue Manager will make every effort to deliver these messages but since they are not necessarily written to disk they will be lost in the case of a system crash or network failure. Clearly with no disk IO involved these messages are much faster (and cheaper) than persistent ones.

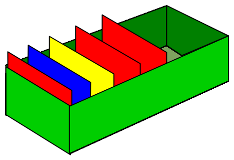


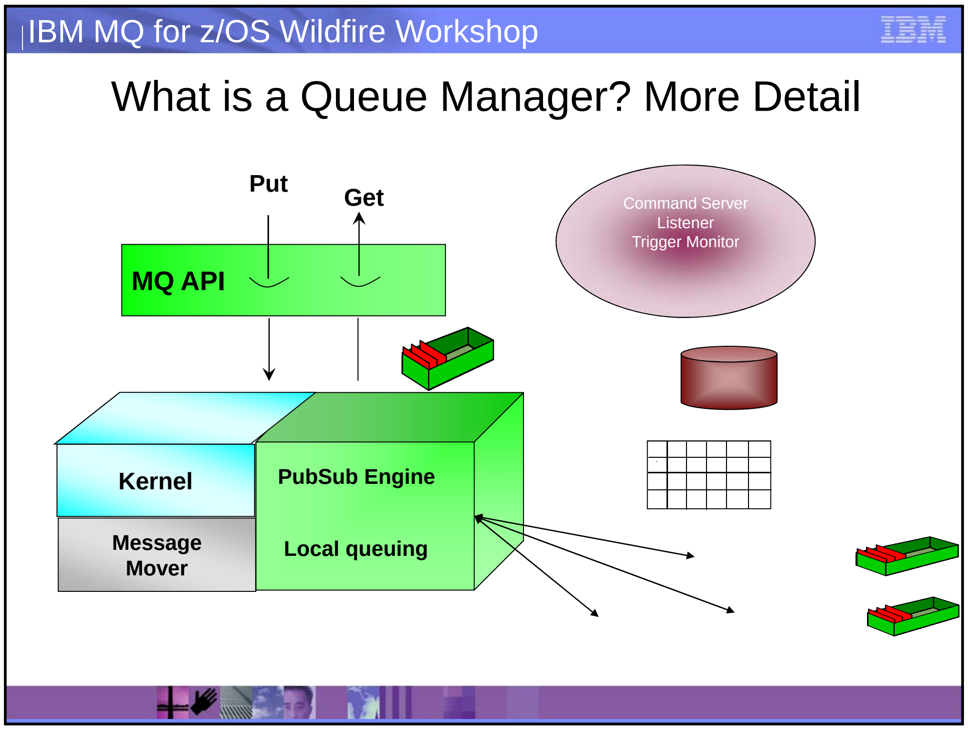
IBM MQ for z/OS Wildfire Workshop

What is a Queue? – More detail

- A queue s a container for messages
  - Various Queue Types
    - Local, Alias, Remote, Model
- Queue creation
  - Predefined
  - Dynamically defined
- Message Access
  - FIFO
  - Direct
  - Selected by Property (V7)
  - Priority
  - Destructive & non-destructive access
- Participates in Transactions

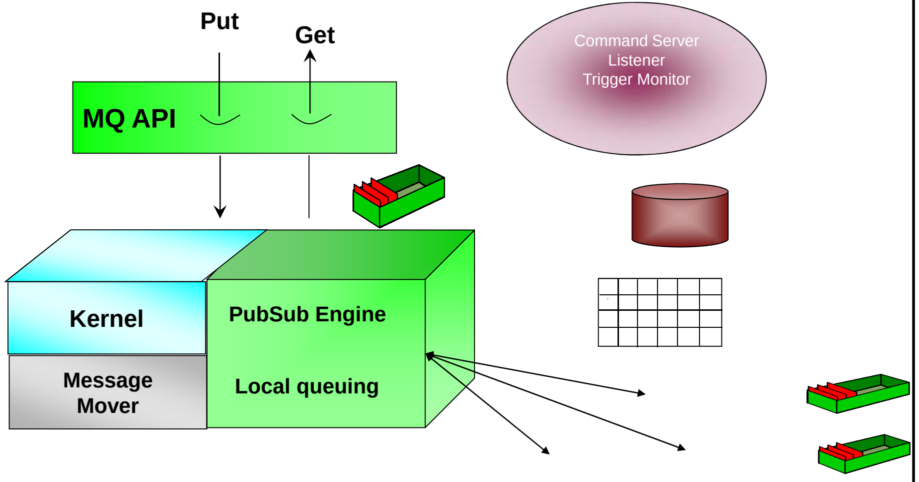
- Parallel access by applications
  - Managed by the queue manager

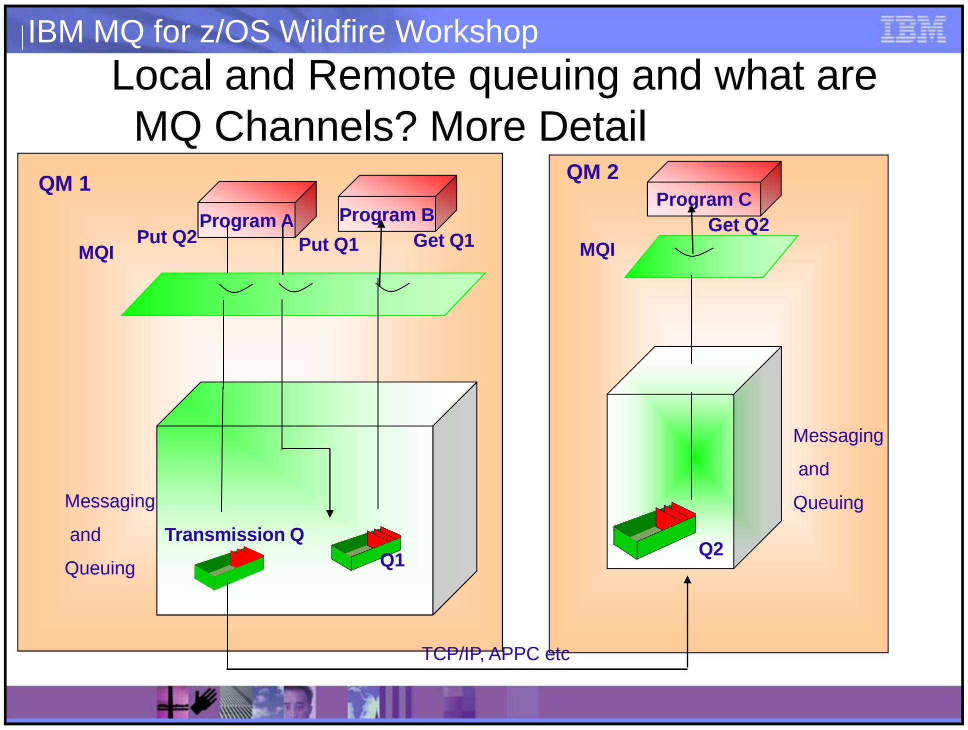




IBM MQ for z/OS Wildfire Workshop

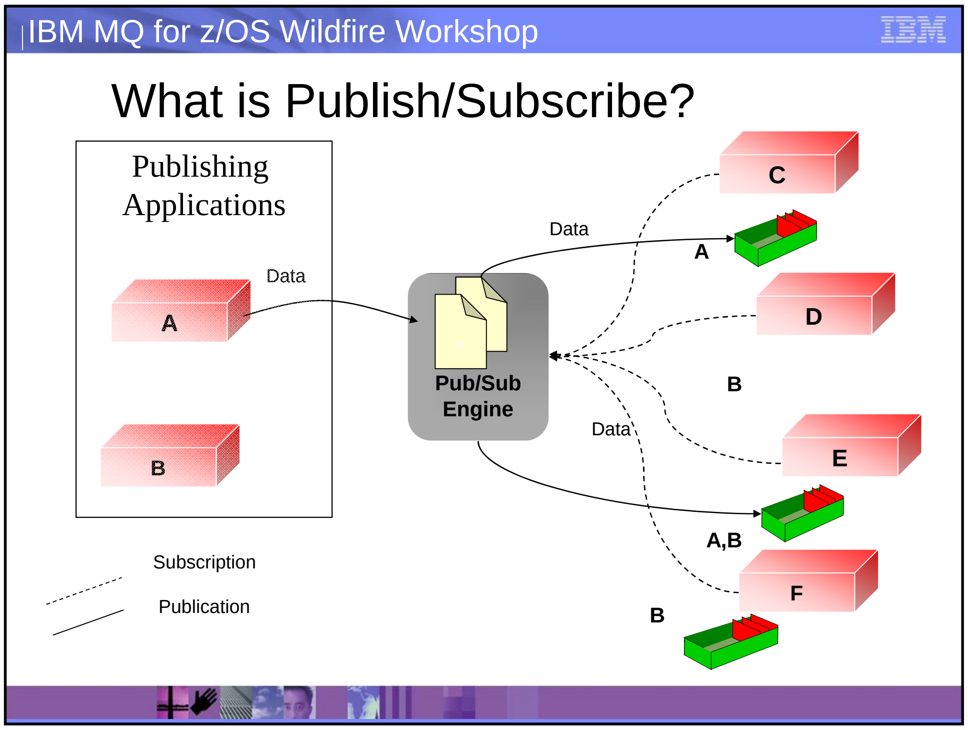
What is a Queue Manager? More Detail





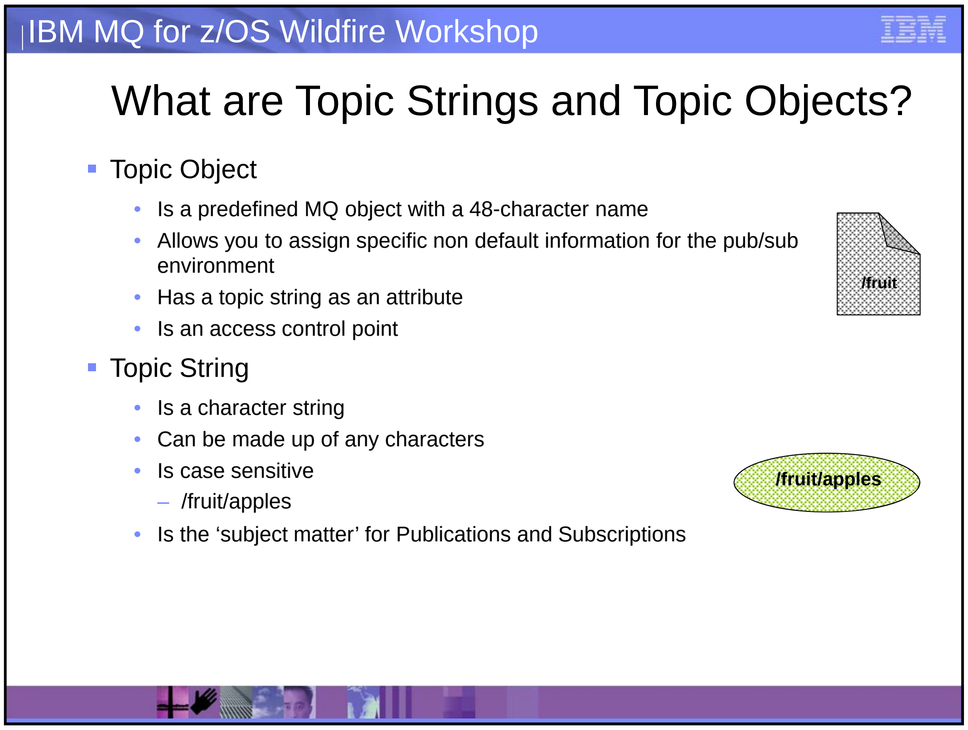
IBM MQ for z/OS Wildfire Workshop

Local and Remote queuing and what are MQ Channels? More Detail



IBM MQ for z/OS Wildfire Workshop

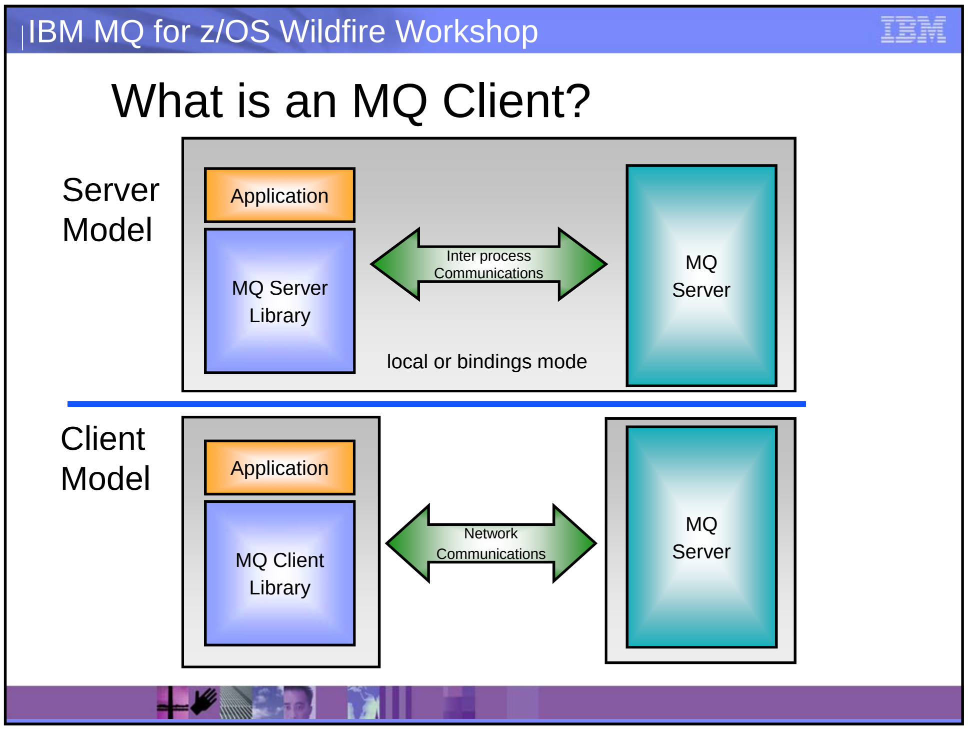
What is Publish/Subscribe?



IBM MQ for z/OS Wildfire Workshop

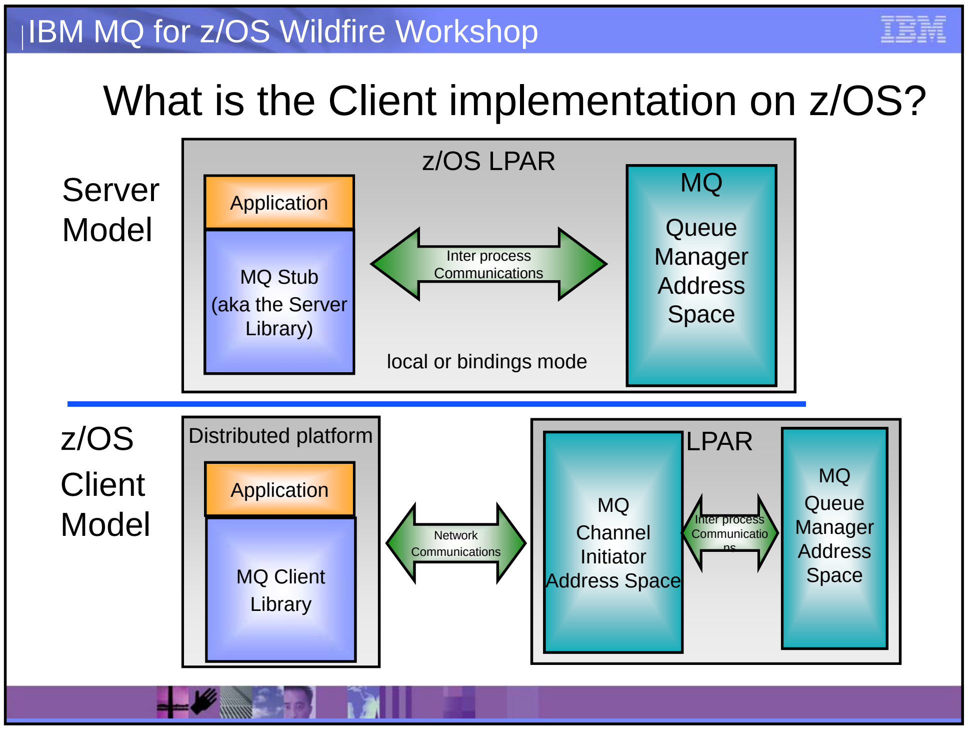
What are Topic Strings and Topic Objects?

- Topic Object
  - Is a predefined MQ object with a 48-character name
  - Allows you to assign specific non default information for the pub/sub environment
  - Has a topic string as an attribute
  - Is an access control point
- Topic String
  - Is a character string
  - Can be made up of any characters
  - Is case sensitive
    - /fruit/apples
  - Is the ‘subject matter’ for Publications and Subscriptions



IBM MQ for z/OS Wildfire Workshop

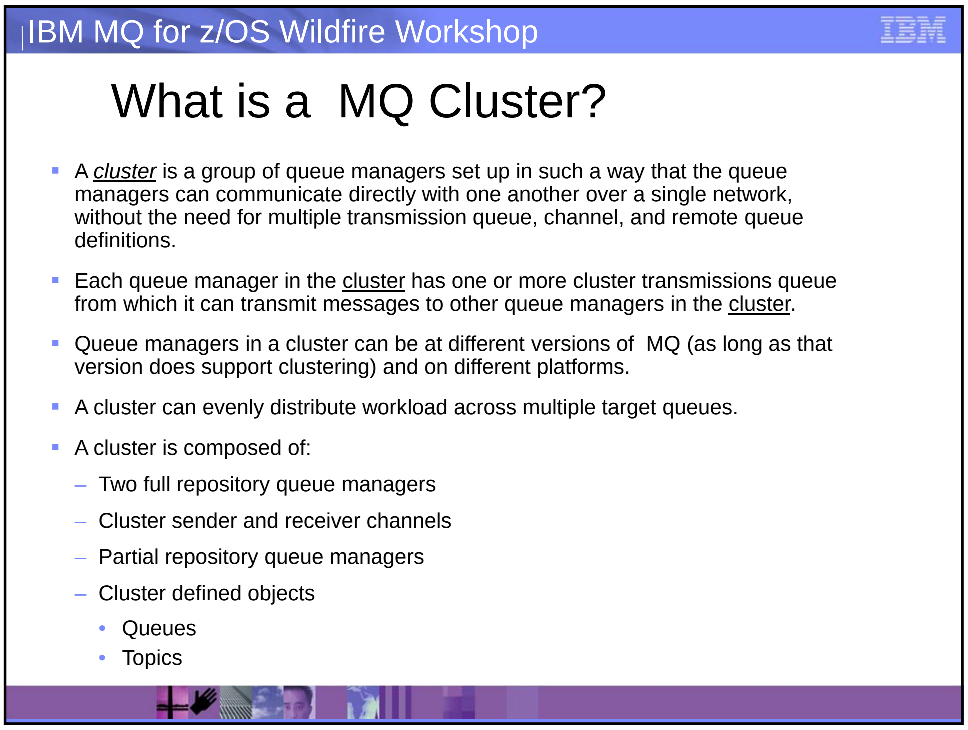
What is an MQ Client?



IBM MQ for z/OS Wildfire Workshop

What is the Client implementation on z/OS?

Server Model
z/OS LPAR
Application
MQ Stub (aka the Server Library)
Inter process Communications
MQ Queue Manager Address Space
local or bindings mode
z/OS Client Model
Distributed platform
Application
MQ Client Library
Network Communications
LPAR
MQ Channel Initiator Address Space
Inter process Communications
MQ Queue Manager Address Space

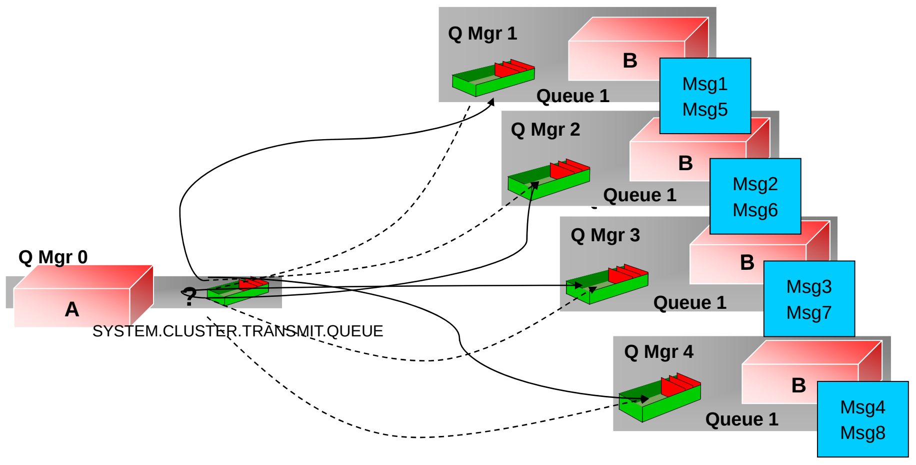


IBM MQ for z/OS Wildfire Workshop

What is a MQ Cluster?

- A *cluster* is a group of queue managers set up in such a way that the queue managers can communicate directly with one another over a single network, without the need for multiple transmission queue, channel, and remote queue definitions.
- Each queue manager in the cluster has one or more cluster transmissions queue from which it can transmit messages to other queue managers in the cluster.
- Queue managers in a cluster can be at different versions of MQ (as long as that version does support clustering) and on different platforms.
- A cluster can evenly distribute workload across multiple target queues.
- A cluster is composed of:
  - Two full repository queue managers
  - Cluster sender and receiver channels
  - Partial repository queue managers
  - Cluster defined objects
    - Queues
    - Topics

How does a cluster normally work?

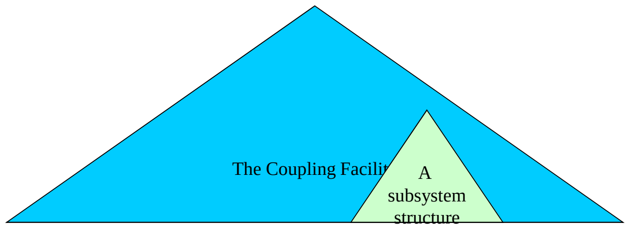


Shared queue terms

- A unique feature to MQ on z/OS, shared queues were designed and built to provide continuous availability for MQ messages.
- In the following foils, the terms used to discuss shared queues are defined

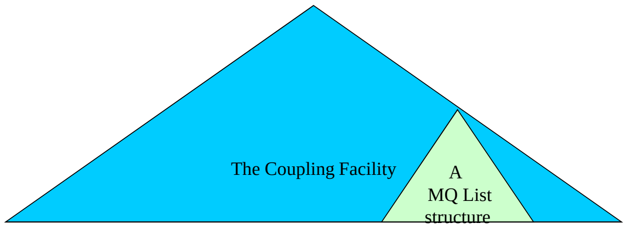
Coupling Facility

- A coupling facility is special hardware and software that allow multiple systems to access the same data. It is unique to z/OS, and is required for a parallel sysplex environment.
- It, and the structures it holds, are typically represented by triangles.



List Structure

- A list structure is a data holding structure in the Coupling Facility used by MQ, IMS and DB2 to hold 'lists' of data. For MQ a single list structure can host up to 512 queues.
- Messages are held on the list structures

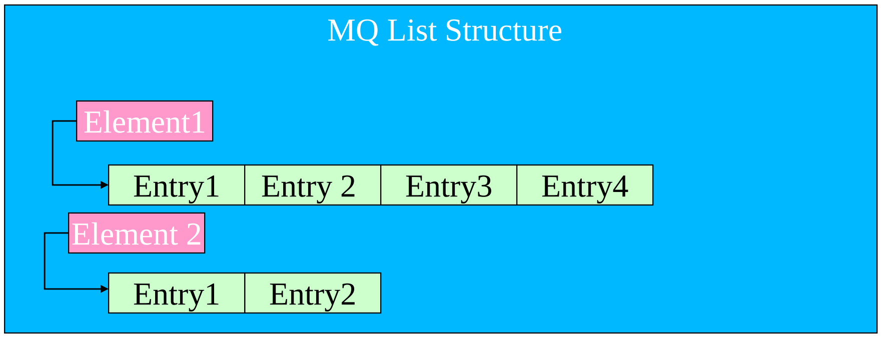


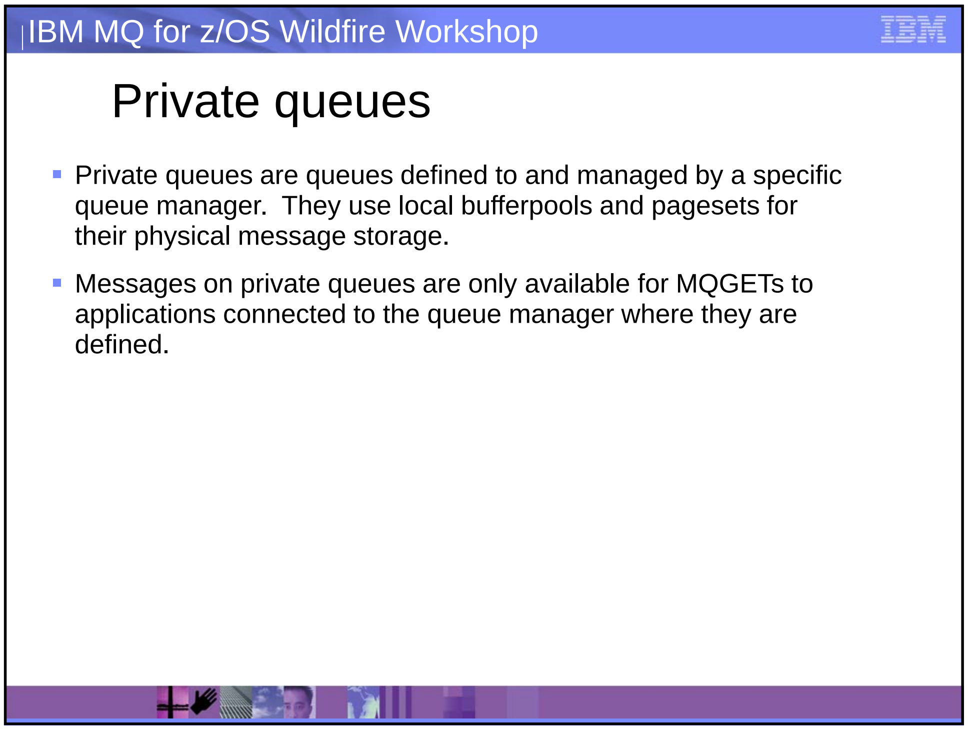
MQ Queue Sharing terms

- Queue Sharing Group
  - A QSG is a logical association of queue managers in a Sysplex. These queue managers are connected to coupling facility list structures and a DB2 Data sharing group. This allows them to share queues and their messages, to treat any queue defined on the CF as if it is local (can do both MQGETs and MQPUTs).
  - There can be up to 32 queue managers in a QSG.
- Shared queues
  - Queues defined on a Coupling Facility structure.
  - Available to every queue manager on the queue shared group as if it is a local queue.
- CFSTRUCT
  - A MQ object that defines the Coupling facility list structure to MQ.

Elements and Entries

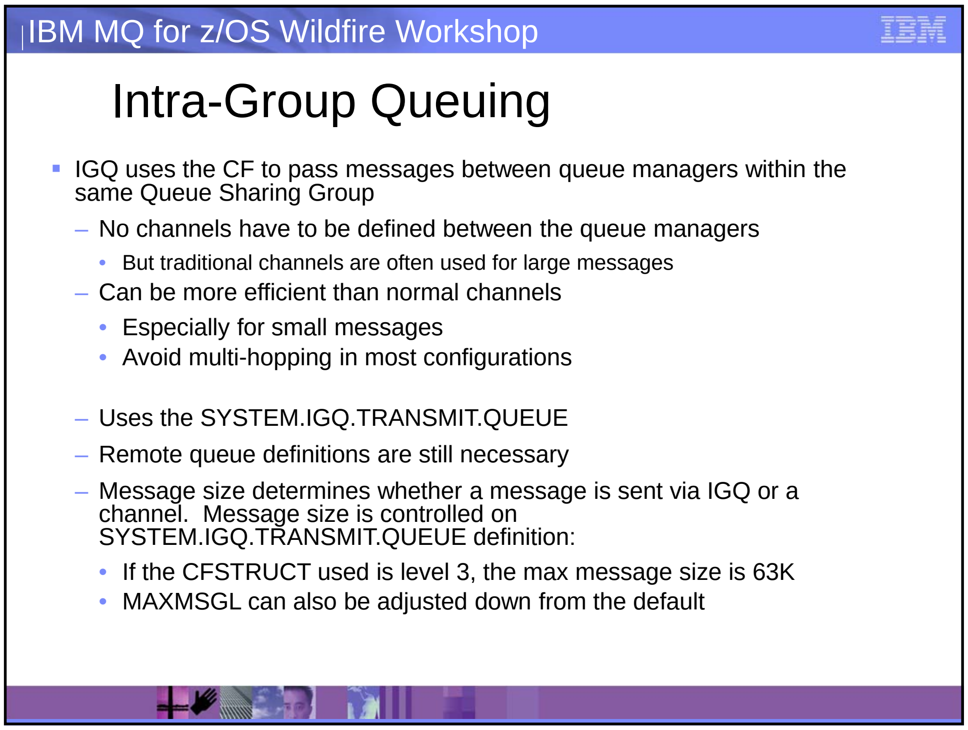
- An element is the anchor of an individual message in the list structure.
- The entries (aka segments) are the 256-byte chunks of the message in the list structure.





Private queues

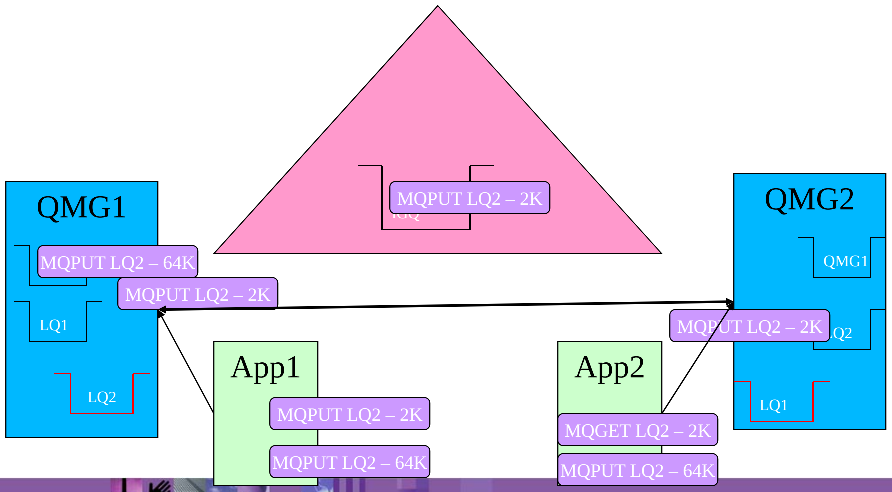
- Private queues are queues defined to and managed by a specific queue manager. They use local bufferpools and pagesets for their physical message storage.
- Messages on private queues are only available for MQGETs to applications connected to the queue manager where they are defined.



Intra-Group Queuing

- IGQ uses the CF to pass messages between queue managers within the same Queue Sharing Group
  - No channels have to be defined between the queue managers
    - But traditional channels are often used for large messages
  - Can be more efficient than normal channels
    - Especially for small messages
    - Avoid multi-hopping in most configurations
  - Uses the SYSTEM.IGQ.TRANSMIT.QUEUE
  - Remote queue definitions are still necessary
  - Message size determines whether a message is sent via IGQ or a channel. Message size is controlled on SYSTEM.IGQ.TRANSMIT.QUEUE definition:
    - If the CFSTRUCT used is level 3, the max message size is 63K
    - MAXMSGL can also be adjusted down from the default

Intra-Group Queuing – What it looks like



Summary

- MQ general terms were defined for use in this session
- MQ Shared queue terms were defined for this session
- Any questions?