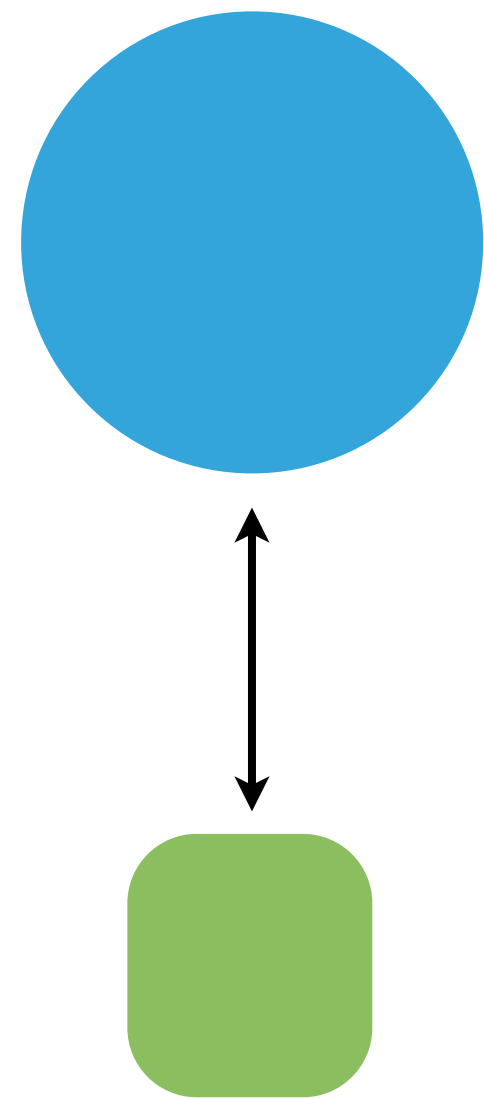# MOBILE APPLICATION DEVELOPMENT

ANDROID (2017)
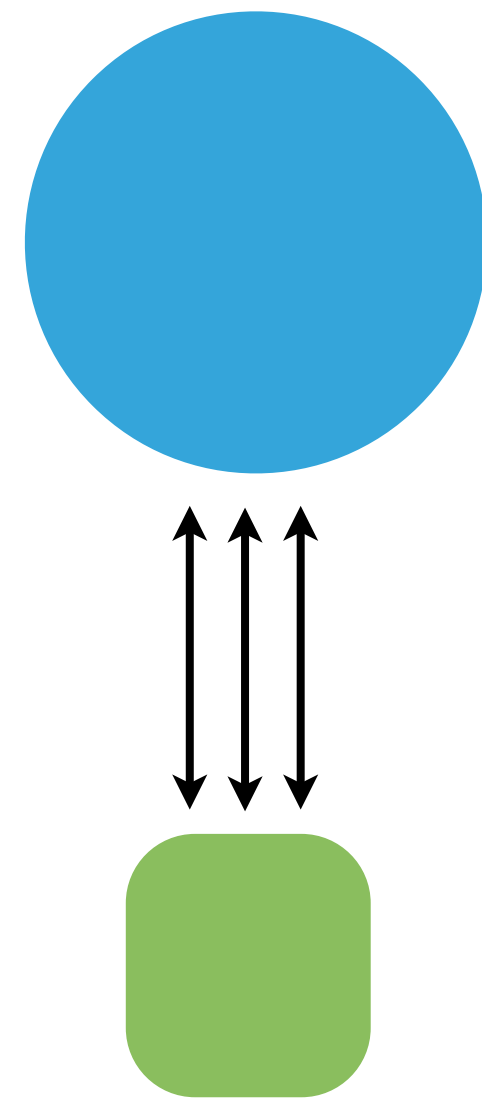
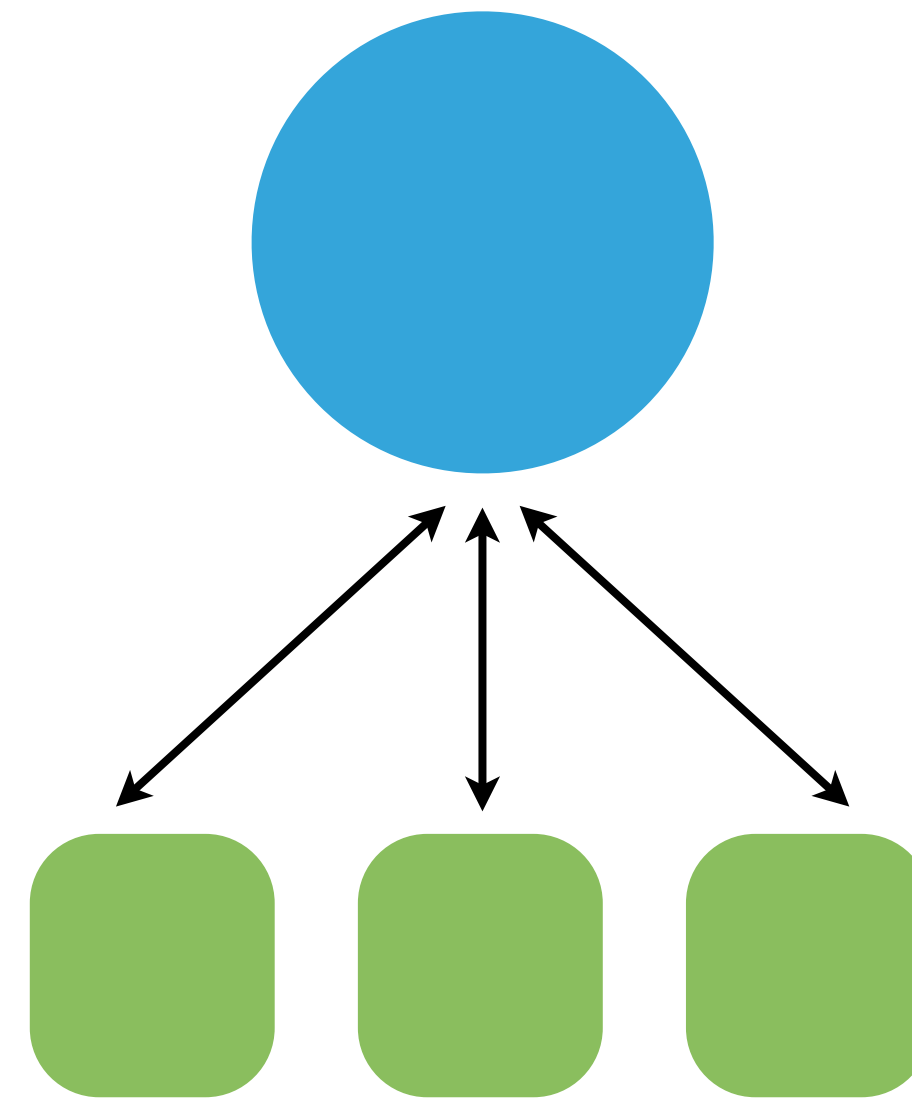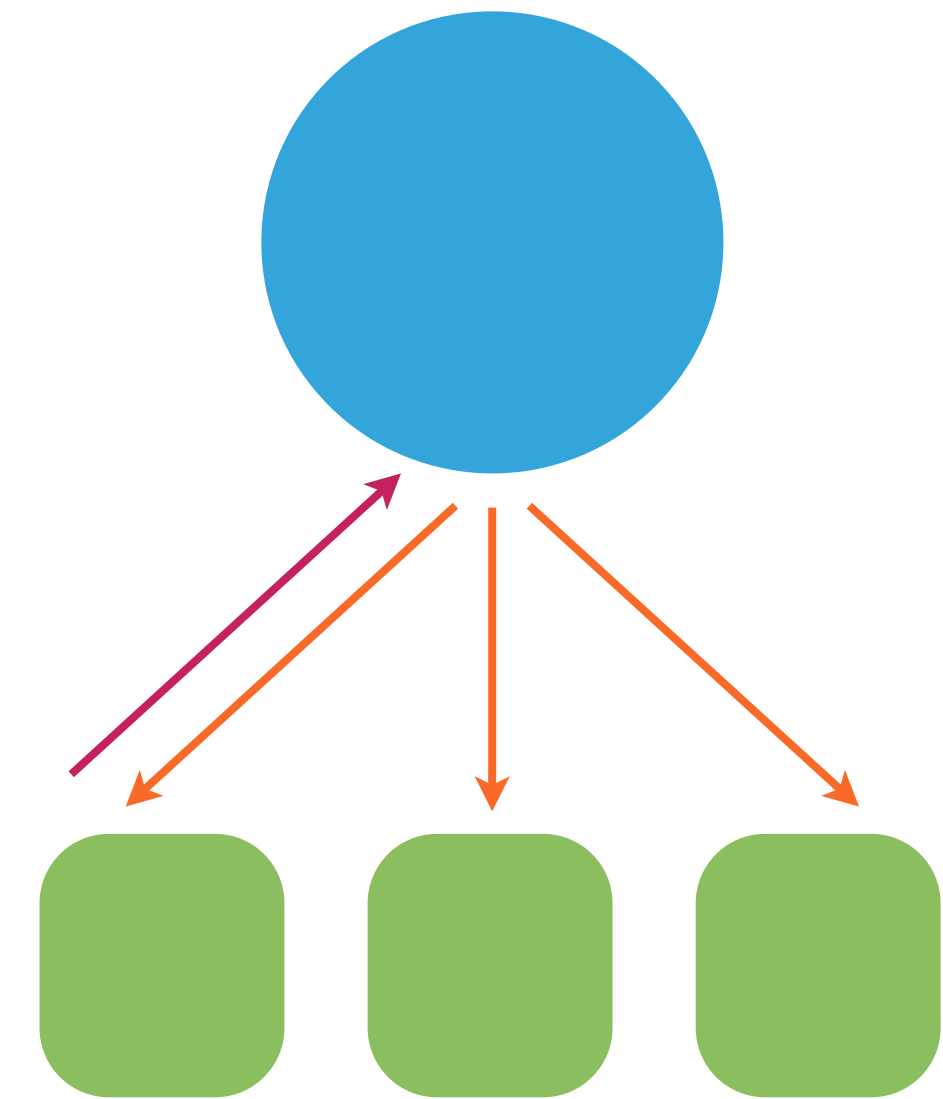## LECTURE 14: MESSAGING

# MESSAGING MECHANISMS

Listener

Interface

Listener
Collection

Broadcast
Manager

LISTENERS

▸ Defined using interfaces, and implemented as objects/classes which implement those interfaces.

▸ Attached to event generators using setters, obtained using getters. Event generators call methods on the listeners when appropriate.

▸ Pros:

▸ Simple, clearly-defined, structured.

▸ Cons:

▸ Only one listener per generator is generally used, and many listeners can become tedious to manage.
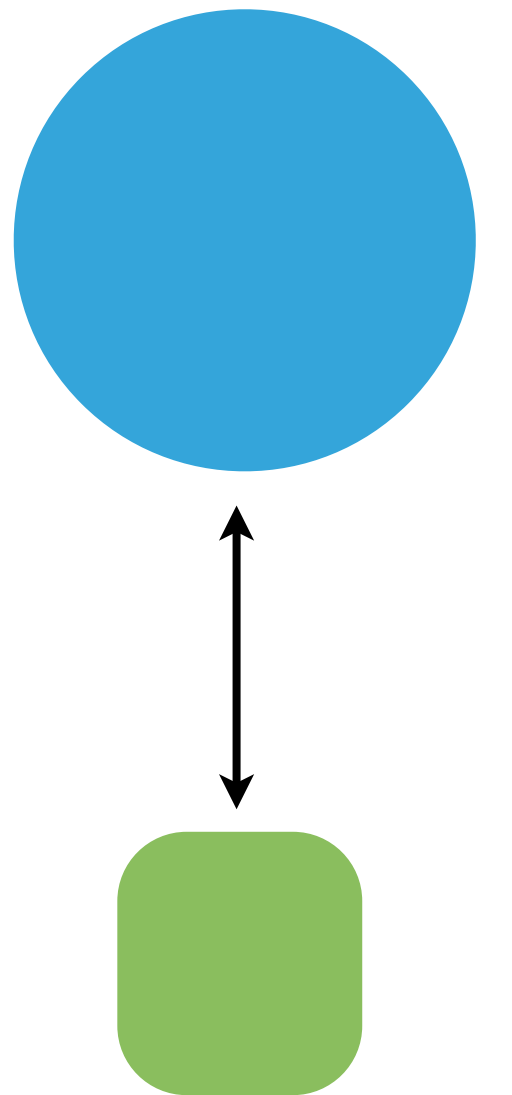
INTERFACES

▸ Essentially define 'listeners with multiple methods'. May not be set up exactly like listener relationships, but function essentially the same way.

▸ Pros:

  ▸ Eliminate the need for multiple listeners for a given event generator.

▸ Cons:

  ▸ Require every 'listener' to implement the entire interface even if only one event is desired.

 ▸ More complex than a listener - can't use a lambda as the 'listener', for example.
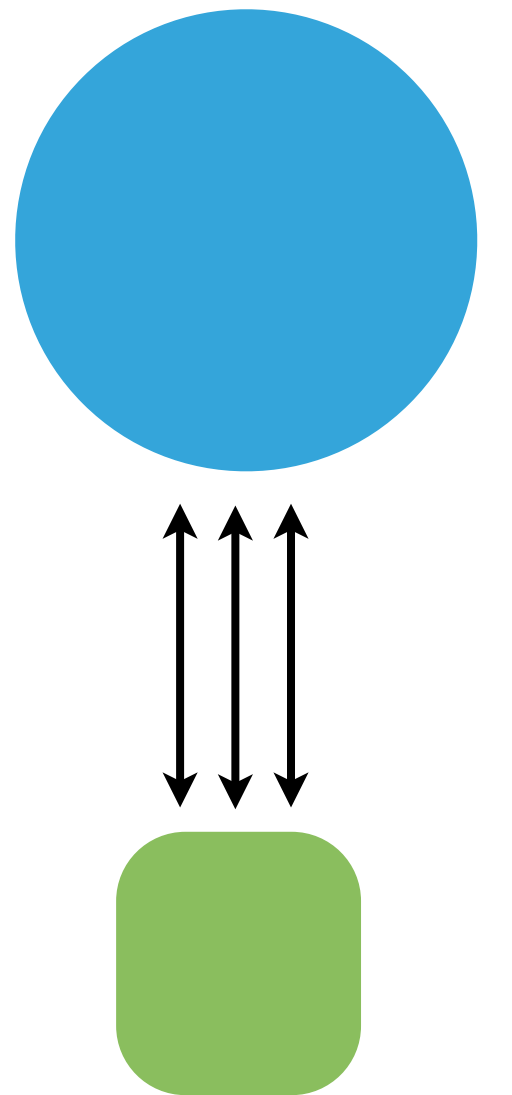
LISTENER COLLECTIONS

▸ Organize listeners into a collection, so that events are sent to all listeners at once, and data can be fetched from all of them.

▸ Pros:

　▸ Allow for complex interactions between many listeners (event generator may request a lot of information in response to events).

▸ Cons:

　▸ Handling a variety of listener responses can be complicated (example: voting).

　▸ Introduce a number of management concerns that simple listeners don't have.
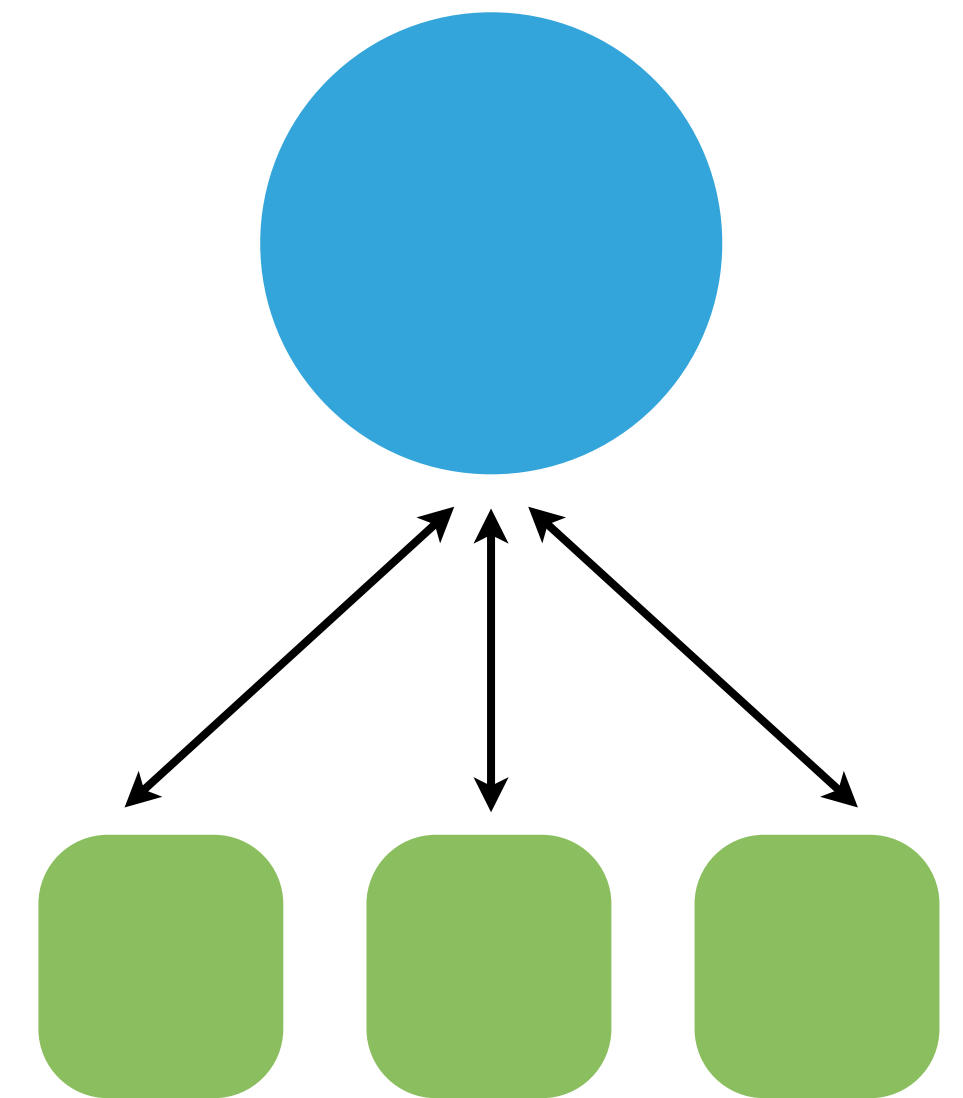
BROADCAST MANAGERS

▶ Establish a publisher-subscriber model where objects/classes register for notifications of events and event generators broadcast events to all interested parties.

▶ Pros:

  ▸ Decouple event generators from subscribers, allowing easy communication across threads.

  ▸ Events may be subscribed to or unsubscribed from at any time.

▶ Cons:

  ▸ Broadcast manager cannot (shouldn't) request information from subscribers.

  ▸ As with listener collections, broadcasting introduces a number of management concerns.
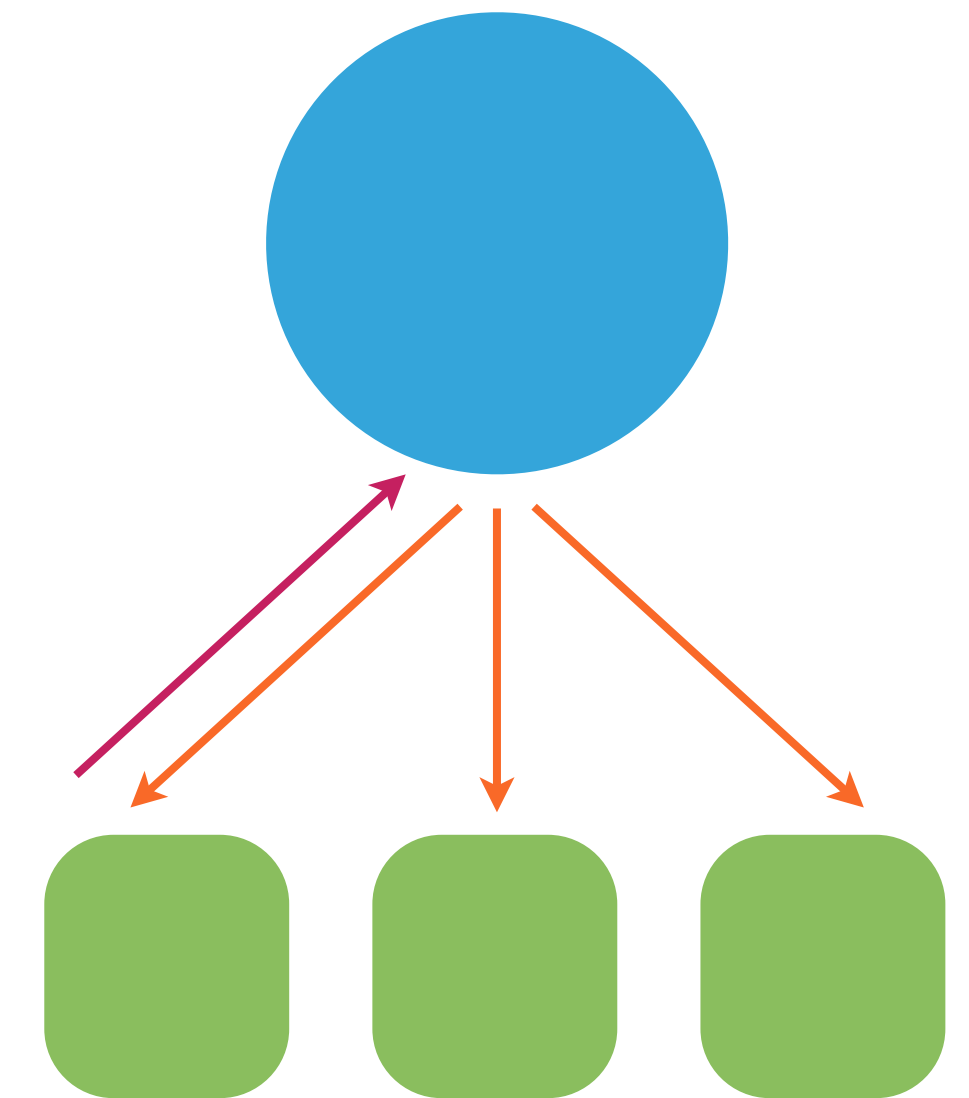
MESSAGING CLEANUP

▶ Listener/subscriber relationships often (though not always) establish strong memory references to the listener/subscriber from the event generator.

▶ If listeners/subscribers are not unregistered when the association between them and the event generator is no longer desired, they will remain associated with the event generator and both will likely never be deallocated.

▶ It is essential to understand the memory management concerns imposed by your messaging strategy and to handle them properly.