EventBus Class

Contains methods for publishing platform events.

Namespace

System

- EventBus Methods

See Also

- *Platform Events Developer Guide*: Publishing Platform Events

EventBus Methods

The following are methods for `EventBus`. All methods are static.

- **getOperationId(result)**
  Returns the event UUID, which identifies a published event message.
- **publish(event)**
  Publishes the given platform event.
- **publish(events)**
  Publishes the given list of platform events.
- **publish(event, callback)**
  Publishes the given platform event using the specified callback. To track asynchronous publish failures, you can implement an Apex publish callback.
- **publish(events, callback)**
  Publishes the given list of platform events using the specified callback. To track asynchronous publish failures, you can implement an Apex publish callback.

getOperationId(result)

Returns the event UUID, which identifies a published event message.

Signature

```
public static String getOperationId(Object result)
```

Parameters

result

Type: Object

The SaveResult that is returned by the `EventBus.publish` call.

Return Value

Type: String

Usage

publish(event)

Publishes the given platform event.

Signature

```
public static Database.SaveResult publish(SObject event)
```

Parameters

event

Type: SObject

An instance of a platform event. For example, an instance of *MyEvent__e*. You must first define your platform event object in your org.

Return Value

Type: Database.SaveResult

The result of publishing the given event. `Database.SaveResult` contains information about whether the operation was successful and the errors encountered. If the `isSuccess()` method returns `true`, the publish request is queued in Salesforce and the event message is published asynchronously. For more information, see High-Volume Platform Event Persistence. If `isSuccess()` returns `false`, the event publish operation resulted in errors, which are returned in the `Database.Error` object. This method doesn't throw an exception due to an unsuccessful publish operation.

`Database.SaveResult` also contains the `Id` system field. The `Id` field value isn't included in the event message delivered to subscribers. It isn't used to identify an event message, and isn't always unique.

This method returns a `System.UnexpectedException` if you attempt to publish an `SObject` that represents an object that isn't a platform event.

Usage

- The platform event message is published either immediately or after a transaction is committed, depending on the publish behavior you set in the platform event definition. For more information, see Platform Event Fields in the *Platform Events Developer Guide*.
- Apex governor limits apply. For events configured with the **Publish After Commit** behavior, each method execution is counted as one DML statement against the Apex DML statement limit. You can check limit usage using the `Limits.getDMLStatements()` method. For events configured with the **Publish Immediately** behavior, each method execution is counted against a separate event publishing limit of 150 `EventBus.publish()` calls. You can check limit usage using the `Limits.getPublishImmediateDML()` method.

publish(events)

Publishes the given list of platform events.

Signature

```
public static List<Database.SaveResult> publish(List<SObject> events)
```

Parameters

events

Type: List<sObject>

A list of platform event instances. For example, a list of *MyEvent__e* objects. You must first define your platform event object in your Salesforce org.

`Database.SaveResult` contains information about whether the operation was successful and the errors encountered. If the `isSuccess()` method returns `true`, the publish request is queued in Salesforce and the event message is published asynchronously. For more information, see High-Volume Platform Event Persistence. If `isSuccess()` returns `false`, the event publish operation resulted in errors, which are returned in the `Database.Error` object. `EventBus.publish()` can publish some passed-in events, even when other events can't be published due to errors. The `EventBus.publish()` method doesn't throw exceptions caused by an unsuccessful publish operation. It's similar in behavior to the Apex `Database.insert` method when called with the partial success option.

`Database.SaveResult` also contains the `Id` system field. The `Id` field value isn't included in the event message delivered to subscribers. It isn't used to identify an event message, and isn't always unique.

If an empty list is passed in for the *events* parameter, no event is published, and an empty `List<Database.SaveResult>` is returned.

This method returns a `System.UnexpectedException` if you attempt to publish a list of type `List<SObject>` that contains objects that aren't platform events.

Usage

- The platform event message is published either immediately or after a transaction is committed, depending on the publish behavior you set in the platform event definition. For more information, see Platform Event Fields in the *Platform Events Developer Guide*.
- Apex governor limits apply. For events configured with the **Publish After Commit** behavior, each method execution is counted as one DML statement against the Apex DML statement limit. You can check limit usage using the `Limits.getDMLStatements()` method. For events configured with the **Publish Immediately** behavior, each method execution is counted against a separate event publishing limit of 150 `EventBus.publish()` calls. You can check limit usage using the `Limits.getPublishImmediateDML()` method.

publish(event, callback)

Publishes the given platform event using the specified callback. To track asynchronous publish failures, you can implement an Apex publish callback.

Signature

```
public static Database.SaveResult publish(SObject event, Object callback)
```

Parameters

event

Type: SObject

An instance of a platform event. For example, an instance of *MyEvent__e*. You must first define your platform event object in your Salesforce org.

callback

Type: Object

An Apex class that implements the EventPublishFailureCallback Interface or EventPublishSuccessCallback Interface.

Return Value

Type: Database.SaveResult

The result of publishing the given event. `Database.SaveResult` contains information about whether the operation was successful and the errors encountered. If the `isSuccess()` method returns `true`, the publish request is queued in Salesforce and the event message is published asynchronously. For more information, see High-Volume Platform Event Persistence. If `isSuccess()` returns `false`,

Usage

- Use this method with Apex publish callbacks. For more information, see Get the Result of Asynchronous Platform Event Publishing with Apex Publish Callbacks in the *Platform Events Developer Guide*.
- The platform event message is published either immediately or after a transaction is committed, depending on the publish behavior you set in the platform event definition. For more information, see Platform Event Fields in the *Platform Events Developer Guide*.
- Apex governor limits apply. For events configured with the **Publish After Commit** behavior, each method execution is counted as one DML statement against the Apex DML statement limit. You can check limit usage using the `Limits.getDMLStatements()` method. For events configured with the **Publish Immediately** behavior, each method execution is counted against a separate event publishing limit of 150 `EventBus.publish()` calls. You can check limit usage using the `Limits.getPublishImmediateDML()` method.

publish(events, callback)

Publishes the given list of platform events using the specified callback. To track asynchronous publish failures, you can implement an Apex publish callback.

Signature

```
public static List<Database.SaveResult> publish(List<SObject> sobjects, Object callback)
```

Parameters

sobjects

Type: List<SObject>

A list of platform event instances. For example, a list of *MyEvent__e* objects. You must first define your platform event object in your Salesforce org.

callback

Type: Object

An Apex class that implements the EventPublishFailureCallback Interface or EventPublishSuccessCallback Interface.

Return Value

Type: List<Database.SaveResult>

A list of results, each corresponding to the result of publishing one event. For each event, `Database.SaveResult` contains information about whether the operation was successful and the errors encountered. If the `isSuccess()` method returns `true`, the publish request is queued in Salesforce and the event message is published asynchronously. For more information, see High-Volume Platform Event Persistence. If `isSuccess()` returns `false`, the event publish operation resulted in errors, which are returned in the `Database.Error` object. `EventBus.publish()` can publish some passed-in events, even when other events can't be published due to errors. The `EventBus.publish()` method doesn't throw exceptions caused by an unsuccessful publish operation. It's similar in behavior to the Apex `Database.insert` method when called with the partial success option.

If an empty list is passed in for the *events* parameter, no event is published, and an empty `List<Database.SaveResult>` is returned.

This method returns a `System.UnexpectedException` if you attempt to publish a list of type `List<SObject>` that contains objects that aren't platform events.

Usage

committed, depending on the publish behavior you set in the platform event definition. For more information, see Platform Event Fields in the *Platform Events Developer Guide*.

- Apex governor limits apply. For events configured with the **Publish After Commit** behavior, each method execution is counted as one DML statement against the Apex DML statement limit. You can check limit usage using the `Limits.getDMLStatements()` method. For events configured with the **Publish Immediately** behavior, each method execution is counted against a separate event publishing limit of 150 `EventBus.publish()` calls. You can check limit usage using the `Limits.getPublishImmediateDML()` method.