Package event

```
import "github.com/emilyhorsman/4zp6/backend/controller/event"
```

Overview ▾

Index ▾

Package files

amqp.go event.go mqtt.go

Variables

```
var (
    // map of registered microcontroller UUIDs
    registered map[string]bool
)
```

func **Start**

```
func Start(s *state.State) error
```

Start will start the main event engine. This is responsible for performing networked I/O with AMQP, MQTT, PostgreSQL. Also responsible for publishing onto Websockets output channel in state.

func **consumeAMQP**

```
func consumeAMQP(s *state.State)
```

consumeAMQP consumes the AMQP output channel found in state. It is responsible for processing incoming AMQP messages.

func **consumeMQTT**

```
func consumeMQTT(s *state.State)
```

consuemMQTT consumes the MQTT output channel found in state. It is responsible for processing incoming MQTT messages.

func **rxConfig**

```
func rxConfig(s *state.State, parts []string, t time.Time, msg
state.AMQPMessage)
```

rxConfig is called when a peripheral processor publishes a message on the "global.config" route. It is responsible for saving the data in the related configuration tables. It also sends a broadcast to all microcontrollers, notifying them of the new configuration.

func **rxPayload**

```
func rxPayload(s *state.State, msg *telemetry.Telemetry, wire
*state.MQTTMessage)
```

rxPayload is called when receiving a payload frame. If the microcontroller is not yet registered, it will send a TX_Request to the microcontroller requesting a registration frame. If the device is registered, it will forward the raw data to AMQP for processing.

func **rxProcessedPayload**

```
func rxProcessedPayload(s *state.State, parts []string, t time.Time, msg
state.AMQPMessage)
```

rxProcessedPayloads is called when a peripheral processor publishes a message on the "data.*.*" route. Is is responsible for saving the data in the "Data" table and publishing data on websocket channel.

func **rxRegistration**

```
func rxRegistration(s *state.State, msg *telemetry.Telemetry, wire
*state.MQTTMessage)
```

rxRegistration is called when receiving a registration frame. It will update the "registration" and "peripheral" tables with data found in the registration message. It then sends all available provisioning profiles back to the microconroller.