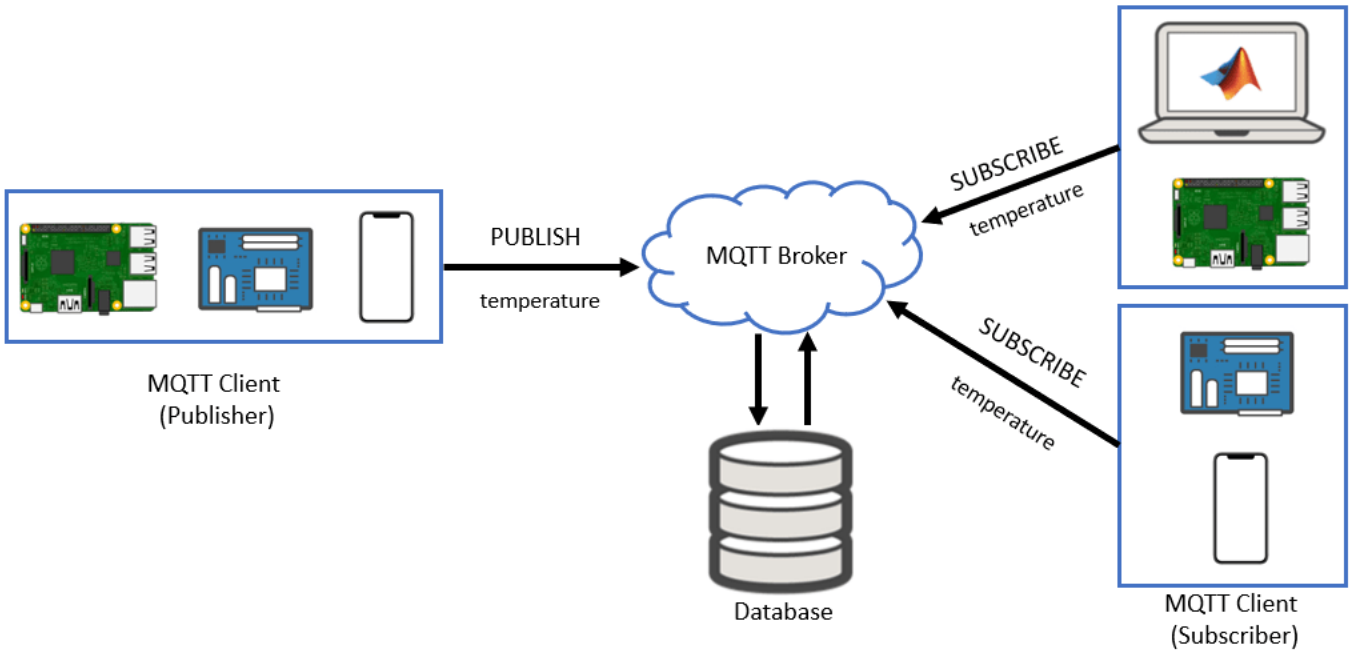
Investigación MQTT



¿Qué es MQTT?

MQTT (Message Queuing Telemetry Transport) es un protocolo de mensajería y eficiente que proporciona una forma sencilla y eficiente de enviar mensajes entre dispositivos conectados a través de una red TCP/IP.

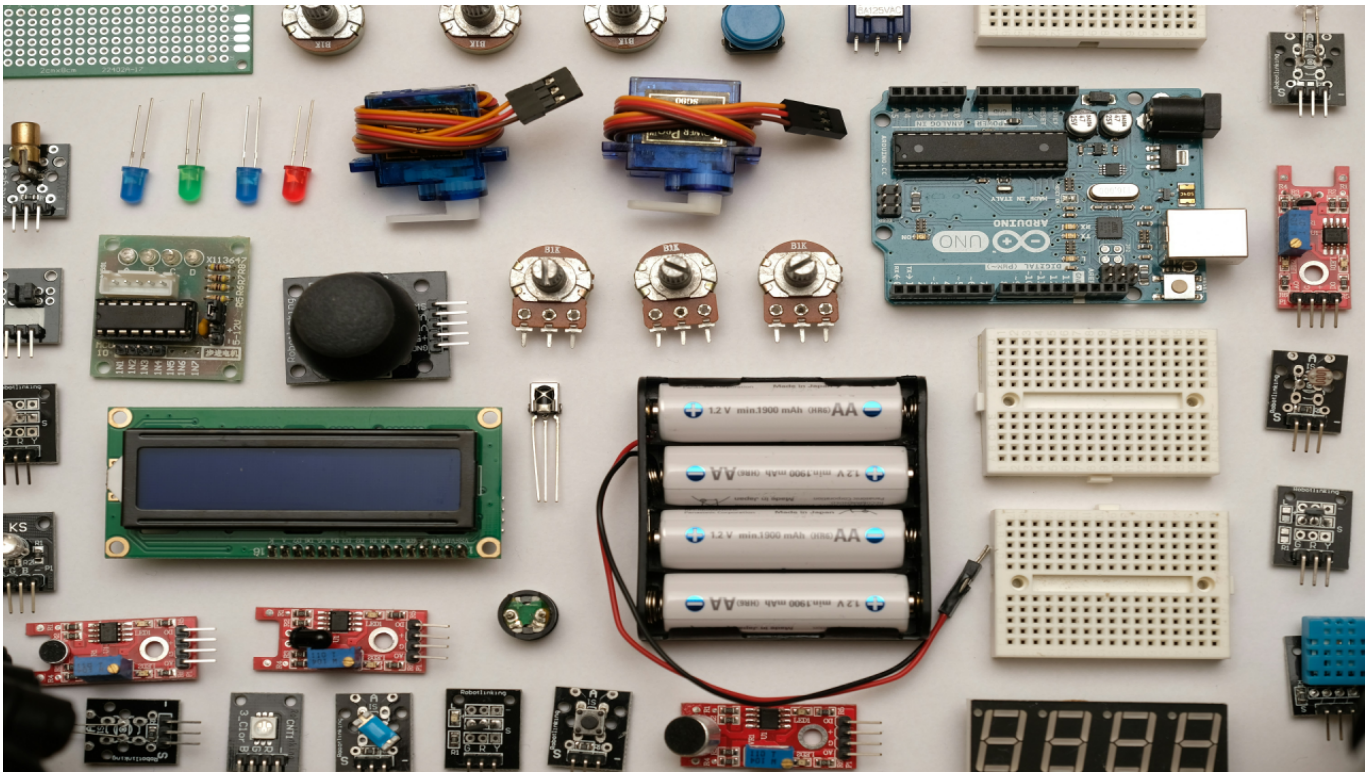
Es un protocolo de mensajería de publicación/suscripción extremadamente simple y liviano, diseñado para dispositivos con ancho de banda limitado y alta latencia. Estos dispositivos pueden ser desde microcontroladores hasta servidores. Suele utilizarse en aplicaciones de IOT.

Características de MQTT

- **Ligero**: MQTT es un protocolo de mensajería ligero y simple que es ideal para dispositivos con ancho de banda limitado y alta latencia.
- **Basado en la publicación/suscripción**: MQTT es un protocolo de mensajería basado en el patrón de diseño de publicación/suscripción. Los clientes se suscriben a ciertos temas y reciben mensajes publicados en esos temas.
- **Confiable**: MQTT es un protocolo confiable que garantiza que los mensajes se entreguen a los clientes una vez y solo una vez. Ofrece tres niveles de calidad de servicio (QoS) para garantizar la entrega de mensajes:
  - **QoS 0 (Entrega mejor esfuerzo)**: El mensaje se envía una vez, pero no hay garantía de que se entregue.
  - **QoS 1 (Entrega una vez)**: El mensaje se envía una vez y se confirma su recepción.
  - **QoS 2**: El mensaje se envía dos veces para garantizar que se entregue exactamente una vez.
- **Seguro**: MQTT admite cifrado y autenticación para proteger los datos.
- **Escalable**: MQTT puede manejar un gran número de dispositivos conectados y mensajes.

Componentes de MQTT

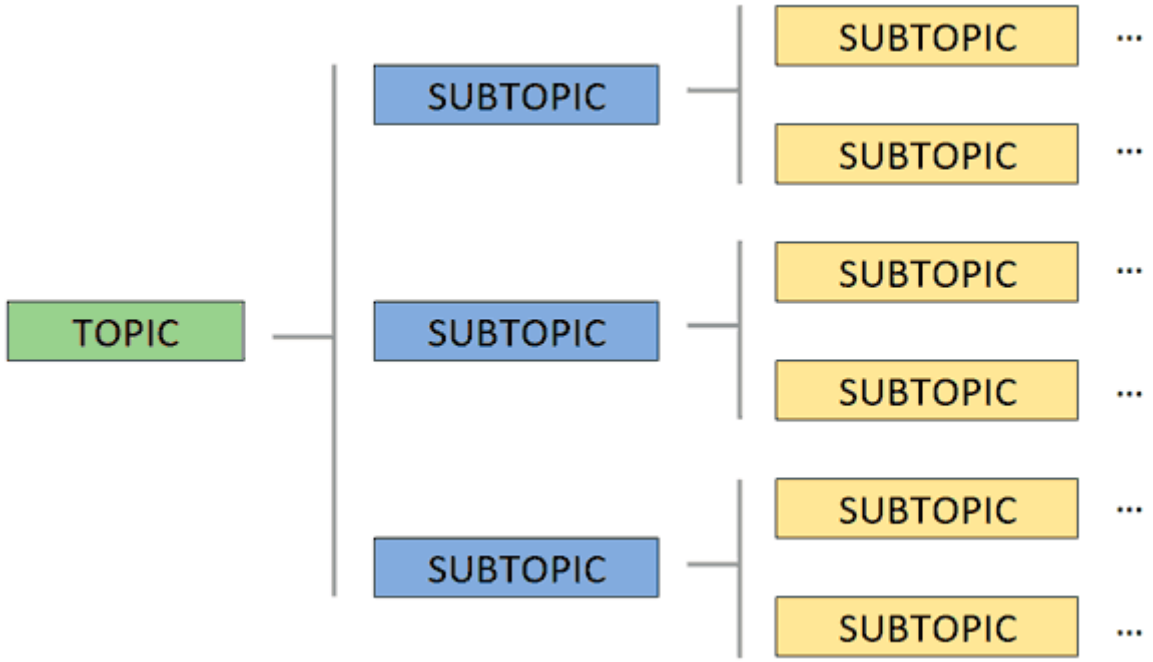
1. **Cliente MQTT**: Un cliente MQTT es cualquier dispositivo que se conecta a un servidor MQTT para enviar o recibir mensajes. Los clientes pueden ser dispositivos de hardware, aplicaciones móviles o aplicaciones de servidor.



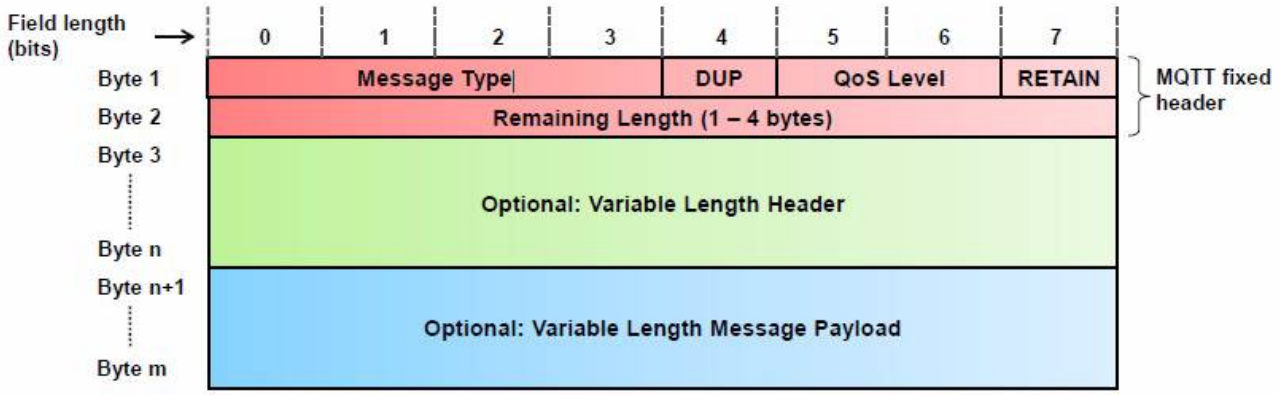
2. **Servidor MQTT**: Un servidor MQTT es un servidor que recibe mensajes de los clientes y los envía a otros clientes. El servidor MQTT es responsable de enrutar los mensajes a los clientes correctos.

3. **Tema**: Un tema es una cadena que se utiliza para filtrar mensajes. Los clientes se suscriben a temas para recibir mensajes publicados en esos temas. Los temas son una forma de categorizar mensajes y permiten a los clientes recibir solo los mensajes que les interesan. Existen también subtemas que se

pueden utilizar para filtrar mensajes de forma más específica mediante el uso de caracteres '/'. Por ejemplo, el tema "sensores/temperatura" podría utilizarse para recibir mensajes de todos los sensores de temperatura.



4. **Mensaje**: Un mensaje es un paquete de datos que se envía de un cliente a otro. Los mensajes pueden contener cualquier tipo de datos, como texto, números, imágenes o archivos. Los mensajes se publican en un tema y se envían a todos los clientes que se han suscrito a ese tema.



Los mensajes de MQTT tienen dos partes principales: el encabezado fijo y la carga útil:

1. **Encabezado fijo**: tiene una longitud fija de 2 bytes y contiene información sobre el tipo de mensaje, la longitud del cuerpo y la calidad de servicio (QoS). Los tipos de mensaje más comunes son:

    - *PUBLISH*: Publica un mensaje en un tema.
    - *SUBSCRIBE*: Se suscribe a un tema.
    - *UNSUBSCRIBE*: Se da de baja de un tema.
    - *PINGREQ*: Solicita una señal de "vivo" del broker.
    - *PINGRESP*: Envía una señal de "vivo" al cliente.
    - *PUBACK*: Confirma la recepción de un mensaje publicado con QoS 1.
    - *PUBREC*: Recibe un mensaje publicado con QoS 2.
    - *PUBREL*: Libera un mensaje publicado con QoS 2.

- *PUBCOMP*: Completa la entrega de un mensaje publicado con QoS 2. También cuenta con un identificador de paquete que se utiliza para identificar los mensajes y una bandera de duplicado que indica si el mensaje es un duplicado.
  Dentro de este campo también se determina el nivel de calidad de servicio (QoS) que se utilizará para enviar el mensaje. Los niveles de QoS son:
- *QoS 0*: Entrega mejor esfuerzo.
- *QoS 1*: Entrega una vez.
- *QoS 2*: Entrega exactamente una vez.

2. **Carga útil**: es la parte del mensaje que contiene los datos reales que se están enviando. La carga útil puede ser cualquier tipo de datos, como texto, números, imágenes o archivos.

Funcionamiento de MQTT

El funcionamiento de MQTT se basa en el patrón de diseño de publicación/suscripción:

1. Un cliente MQTT establecer una conexión con el agente MQTT.
2. Una vez conectado, el cliente puede publicar mensajes, suscribirse a mensajes específicos o hacer ambas cosas.
3. Cuando el agente MQTT recibe un mensaje, lo reenvía a los suscriptores que están interesados.

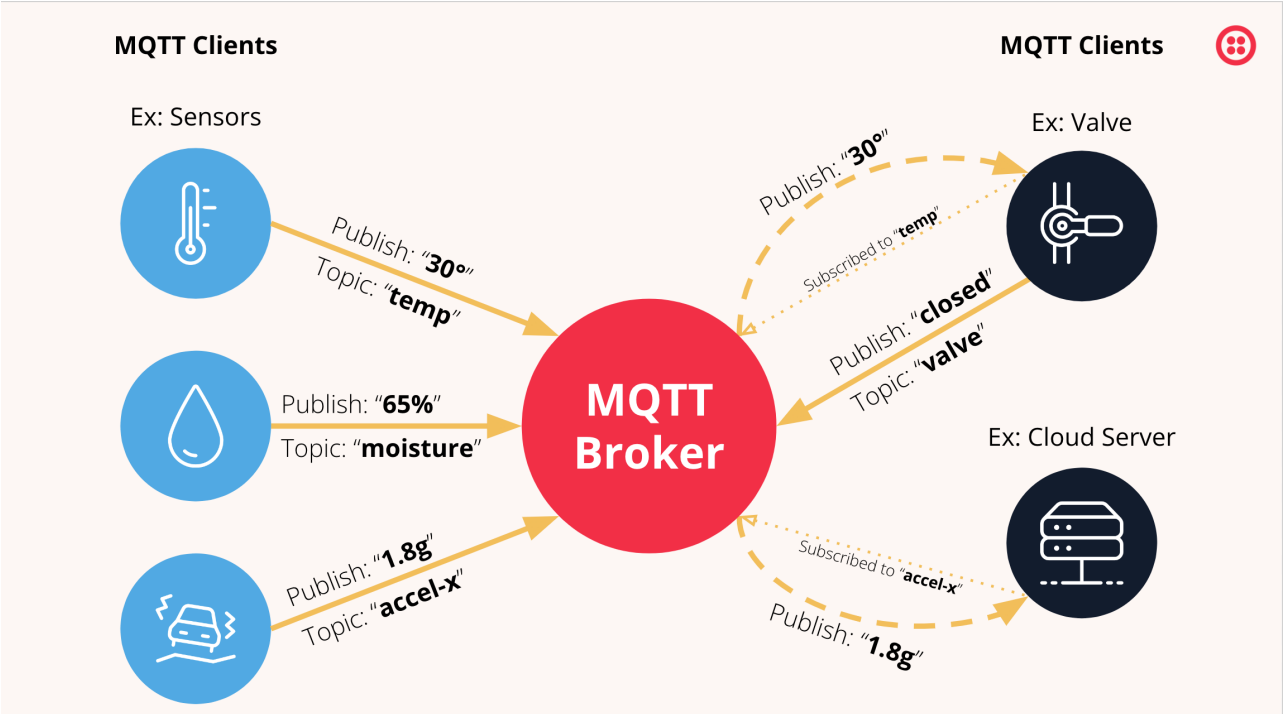
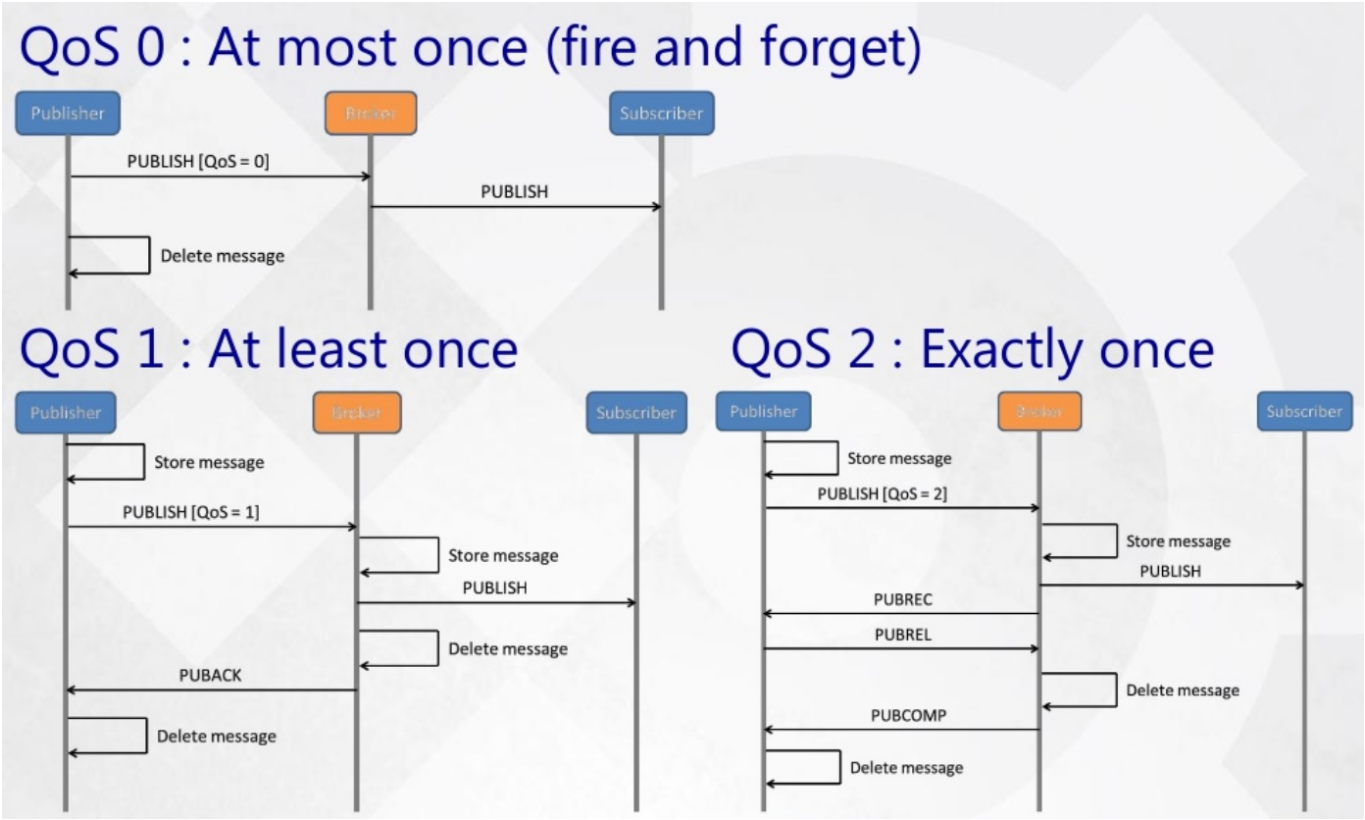


Diagrama QoS



Requerimientos del servidor

Para implementar un servidor MQTT, se requiere un servidor MQTT que actúe como intermediario entre los clientes MQTT. El servidor MQTT es responsable de enrutar los mensajes a los clientes correctos y garantizar que los mensajes se entreguen de manera confiable. Algunos servidores MQTT populares incluyen:

- **Mosquitto**: Mosquitto es un servidor MQTT de código abierto que es ligero y fácil de instalar. Es compatible con los niveles de calidad de servicio (QoS) 0, 1 y 2 y ofrece cifrado y autenticación para proteger los datos.
- **HiveMQ**: HiveMQ es un servidor MQTT de alto rendimiento que es escalable y confiable. Es compatible con los niveles de calidad de servicio (QoS) 0, 1 y 2 y ofrece cifrado y autenticación para proteger los datos.
- **EMQ**: EMQ es un servidor MQTT de código abierto que es escalable y confiable. Es compatible con los niveles de calidad de servicio (QoS) 0, 1 y 2 y ofrece cifrado y autenticación para proteger los datos.

Requerimientos de los clientes

Para implementar un cliente MQTT, se requiere un cliente MQTT que se conecte a un servidor MQTT para enviar o recibir mensajes. Los clientes MQTT pueden ser dispositivos de hardware, aplicaciones móviles o aplicaciones de servidor. Algunos clientes MQTT populares incluyen:

- **Paho**: Paho es una biblioteca MQTT de código abierto que es compatible con una amplia variedad de lenguajes de programación, incluidos Python, Java, JavaScript y C. Paho es fácil de usar y ofrece una API simple para enviar y recibir mensajes MQTT.
- **Eclipse IoT**: Eclipse IoT es una plataforma de código abierto que incluye una variedad de herramientas y bibliotecas para desarrollar aplicaciones de IoT. Eclipse IoT incluye una biblioteca

MQTT de código abierto que es fácil de usar y ofrece una API simple para enviar y recibir mensajes MQTT.

- **MQTT.js**: MQTT.js es una biblioteca MQTT de código abierto para JavaScript que es fácil de usar y ofrece una API simple para enviar y recibir mensajes MQTT. MQTT.js es compatible con los navegadores web y Node.js.

Referencias

- Anónimo. (2022). *MQTT*. MQTT. Recuperado de: https://mqtt.org/
- Anónimo. (*s.f.*).*Publish MQTT Messages and Subscribe to Message Topics*. MathWorks. https://es.mathworks.com/help/simulink/supportpkg/android_ug/publish-and-subscribe-to-mqtt-messages.html
- Anónimo. (*s.f.*). *¿Qué es MQTT?*. AWS Amazon. https://aws.amazon.com/es/what-is/mqtt/
- Belegu, B. (2023). *Setting up an MQTT Server*. Medium. Recuperado de: https://medium.com/@besnikbelegu/setting-up-an-mqtt-server-part-1-87b7bd7d30fd
- Egli, P. R. (2013). *MQTT - MQ Telemetry Transport for Message Queueing*. Indigoo, SlideShare. Recuperado de: https://www.slideshare.net/slideshow/mq-telemetry-transport/25668863#9
- Rusnak, O. (2022). *How the MQTT Client and Broker Connection Works*. Cedalo. Recuperado de: https://cedalo.com/blog/mqtt-connection-beginners-guide/