METODOS DE ENTRADA WEAR OS

Desarrollo Para Dispositivos Móviles

ITIC. Iván E. García Quintero

Gilberto Rodríguez Ramírez
191004

Crear editores de métodos de entrada en Wear

Wear OS admite métodos de entrada más allá de la voz, al ampliar el marco del editor de métodos de entrada (IME) de Android. El marco IME brinda soporte para teclados virtuales en pantalla que permiten a los usuarios ingresar texto en forma de clics de teclas, escritura a mano o gestos.

Los usuarios de Wear OS pueden elegir entre varias opciones de entrada desde Entrada remota. Estas opciones incluyen:

- Dictado
- emoticonos
- Respuestas preparadas
- Respuesta inteligente
- IME predeterminado

Cómo crear un método de entrada

Un editor de método de entrada (IME) es un control para el usuario que le permite ingresar texto. Android proporciona un marco de método de entrada extensible que permite que las aplicaciones ofrezcan a los usuarios métodos de entrada alternativos, como teclados en pantalla o incluso entrada de voz. Después de instalar los IME deseados, un usuario puede seleccionar cuál usar desde la configuración y utilizarlo en todo el sistema; solo se puede habilitar un IME a la vez.

Cómo declarar componentes de IME en el manifiesto

El archivo de manifiesto de la aplicación debe declarar el servicio, solicitar los permisos necesarios, proporcionar un filtro de intentos que coincida con el método action.view.InputMethodde la acción y ofrecer metadatos que definan las características del IME. Además, para proporcionar una interfaz

de configuración que permita que el usuario modifique el comportamiento del IME, puede definir una actividad de "configuración" que se pueda iniciar desde Configuración del sistema.

```
<!-- Declares the input method service -->
<service android:name="FastInputIME"
    android:label="@string/fast_input_label"
    android:permission="android.permission.BIND_INPUT_METHOD">
    <intent-filter>
        <action android:name="android.view.InputMethod" />
    </intent-filter>
    <meta-data android:name="android.view.im"
               android:resource="@xml/method" />
</service>
```

El siguiente fragmento declara la actividad de configuración para el IME. Tiene un filtro de intents para ACTION_MAINque indica que esta actividad es el punto de entrada principal para la aplicación del IME:

```
<!-- Optional: an activity for controlling the IME settings -->
<activity android:name="FastInputIMESettings"
    android:label="@string/fast_input_settings">
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
    </intent-filter>
</activity>
```

Entrada rotativa

Algunos dispositivos Wear OS contienen un botón lateral giratorio físico . Cuando el usuario gira el botón, la vista actual de su aplicación debe desplazarse hacia arriba o hacia abajo. Este tipo de entrada se llama entrada rotativa .

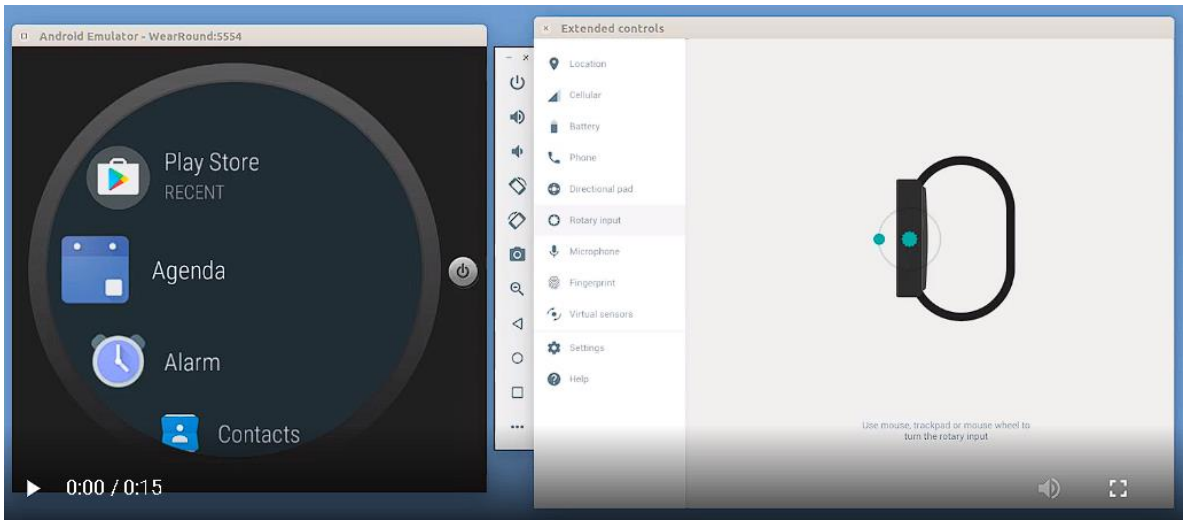
Muchos contenedores desplazables, como ScrollView, ListView, HorizontalScrollViewy WearableRecyclerView, admiten entradas rotativas sin necesidad de ningún código específico de Wear OS, si estos contenedores tienen foco. Tener foco es un requisito previo importante, porque en Android 9 (nivel de API 28) y versiones posteriores, las vistas no reciben foco de forma implícita.

El siguiente fragmento de código muestra cómo usar MotionEvent, InputDeviceCompaty ViewConfigurationCompat para agregar un desplazamiento personalizado a su vista:

```
myView.setOnGenericMotionListener(new View.OnGenericMotionListener() {
  @Override
  public boolean onGenericMotion(View v, MotionEvent ev) {
    if (ev.getAction() == MotionEvent.ACTION_SCROLL &&
        ev.isFromSource(InputDeviceCompat.SOURCE_ROTARY_ENCODER)
    ) {
      // Don't forget the negation here
      float delta = -ev.getAxisValue(MotionEventCompat.AXIS_SCROLL) *
          ViewConfigurationCompat.getScaledVerticalScrollFactor(
              ViewConfiguration.get(context), context
          );

      // Swap these axes to scroll horizontally instead
      v.scrollBy(0, Math.round(delta));

      return true;
    }
    return false;
  }
});
```



Gboard for Wear OS

Si no lo sabía, Wear OS ofrece un teclado incorporado para escribir respuestas a mensajes sin tener que alcanzar su teléfono. Puede optar por confiar en la autocorrección y escribir manualmente su respuesta, también puede escribir con gestos si lo prefiere, o simplemente puede ingresar su texto por entrada de voz.

Se han mejorado las sugerencias al permitirle desplazarse por ellas más fácilmente. Una nueva 'Pantalla de vista previa' le permitirá obtener una vista previa de todo el mensaje antes de enviarlo. La compatibilidad con varios idiomas es una nueva característica de Gboard para Wear OS. Un nuevo botón de entrada le permitirá cambiar a otro idioma habilitado en Gboard.

Voice input

Cada dispositivo Wear OS viene con un micrófono, por lo que los usuarios pueden usar su voz para interactuar con el dispositivo. Puede dividirlos en tres tipos de interacciones:

- Grabar audio
- Obtenga entrada de voz de forma libre
- Acciones de voz

Grabar audio

La grabación de audio en un dispositivo Wear OS funciona de la misma manera que en un teléfono. Consulte la documentación de MediaRecorder para obtener más información sobre cómo grabar audio en Android. También puede ver una implementación de muestra en la muestra de Wear Speaker en Github.

Obtenga entrada de voz de forma libre

Llame a la actividad del reconocedor de voz incorporado del sistema para obtener la entrada de voz de los usuarios. Utilice la entrada de voz para enviar mensajes o realizar búsquedas.

En su aplicación, llame startActivityForResult() usando la ACTION_RECOGNIZE_SPEECH acción. Esto inicia la actividad de reconocimiento de voz y luego puede manejar el resultado en formato onActivityResult().

El siguiente ejemplo de código muestra cómo iniciar y manejar una actividad de reconocimiento de voz.

```
private static final int SPEECH_REQUEST_CODE = 0;

// Create an intent that can start the Speech Recognizer activity
private void displaySpeechRecognizer() {
    Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
            RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
// This starts the activity and populates the intent with the speech text.
    startActivityForResult(intent, SPEECH_REQUEST_CODE);
}

// This callback is invoked when the Speech Recognizer returns.
// This is where you process the intent and extract the speech text from the intent.
@Override
protected void onActivityResult(int requestCode, int resultCode,
        Intent data) {
    if (requestCode == SPEECH_REQUEST_CODE && resultCode == RESULT_OK) {
        List<String> results = data.getStringArrayListExtra(
                RecognizerIntent.EXTRA_RESULTS);
        String spokenText = results.get(0);
        // Do something with spokenText.
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

Referencias.

Create input method editors on Wear. Developers. https://developer.android.com/training/wearables/user-input/wear-ime

Rotary input. Developers. https://developer.android.com/training/wearables/user-input/rotary-input#java

Gboard para Wear OS actualizado con entrada de texto mejorada, compatibilidad con varios idiomas y una nueva apariencia. GSMArena. https://www.gsmarena.com/gboard_for_wear_os_updated_with_enhanced_text_input_multilanguage_support_and_a_new_look-news-48994.php

Voice input. Developers. https://developer.android.com/training/wearables/user-input/voice