

Захват видео с собственного программного источника

Разработка

В сборке Android SDK [1.1.0.26](#) добавлена возможность подключить собственную программную реализацию камеры для захвата изображения. Для этого необходимо:

1. Разработать Java класс, реализующий интерфейс [CameraVideoCapturer](#)
2. Импортировать в приложение модули

```
import org.webrtc.CameraVideoCapturer;
import com.flashphoner.fpwcsapi.camera.CameraCapturerFactory;
import com.flashphoner.fpwcsapi.camera.CustomCameraCapturerOptions;
import com.flashphoner.fpwcsapi.camera.CustomCameras;
```

3. Подготовить объект `CustomCameraCapturerOptions`

```
private CustomCameraCapturerOptions createCustomCameraCapturerOptions() {
    return new CustomCameraCapturerOptions() {

        private String cameraName;
        private CameraVideoCapturer.CameraEventsHandler eventsHandler;
        private boolean captureToTexture;

        @Override
        public Class<?>[] getCameraConstructorArgsTypes() {
            return new Class<?>[]{String.class,
                CameraVideoCapturer.CameraEventsHandler.class, boolean.class};
        }

        @Override
        public Object[] getCameraConstructorArgs() {
            return new Object[]{cameraName, eventsHandler,
                captureToTexture};
        }

        @Override
        public void setCameraName(String cameraName) {
            this.cameraName = cameraName;
        }

        @Override
        public void setEventsHandler(CameraVideoCapturer.CameraEventsHandler
            eventsHandler) {
```

```

        this.eventsHandler = eventsHandler;
    }

    @Override
    public void setCaptureToTexture(boolean captureToTexture) {
        this.captureToTexture = captureToTexture;
    }

    // Use your custom capturer class name here
    @Override
    public String getCameraClassName() {
        return your.custom.CameraCapturer;
    }

    @Override
    public Class<?>[] getEnumeratorConstructorArgsTypes() {
        return new Class[0];
    }

    @Override
    public Object[] getEnumeratorConstructorArgs() {
        return new Object[0];
    }

    // Use your custom capturer enumerator name here
    @Override
    public String getEnumeratorClassName() {
        return your.custom.CameraEnumerator;
    }
};
}

```

4. В приложении перед публикацией потока выбрать собственную камеру

```

CameraCapturerFactory.getInstance().setCustomCameraCapturerOptions(createCustomCameraCapturerOptions());
CameraCapturerFactory.getInstance().setCameraType(CameraCapturerFactory.CameraType.CUSTOM);

```

Пример использования

Используем собственную реализацию CameraVideoCapturer для доступа к вспышке:

1. Создание объекта `CustomCameraCapturerOptions`

code

```

private CustomCameraCapturerOptions createCustomCameraCapturerOptions() {
    return new CustomCameraCapturerOptions() {

        private String cameraName;
        private CameraVideoCapturer.CameraEventsHandler eventsHandler;
        private boolean captureToTexture;
    };
}

```

```

        @Override
        public Class<?>[] getCameraConstructorArgsTypes() {
            return new Class<?>[]{String.class,
CameraVideoCapturer.CameraEventsHandler.class, boolean.class};
        }

        @Override
        public Object[] getCameraConstructorArgs() {
            return new Object[]{cameraName, eventsHandler,
captureToTexture};
        }

        @Override
        public void setCameraName(String cameraName) {
            this.cameraName = cameraName;
        }

        @Override
        public void setEventsHandler(CameraVideoCapturer.CameraEventsHandler
eventsHandler) {
            this.eventsHandler = eventsHandler;
        }

        @Override
        public void setCaptureToTexture(boolean captureToTexture) {
            this.captureToTexture = captureToTexture;
        }

        // Using org.webrtc.FlashlightCameraCapturer to access flashlight
hidden controls.
        @Override
        public String getCameraClassName() {
            return CustomCameras.FLASHLIGHT_CAMERA_CAPTURER;
        }

        @Override
        public Class<?>[] getEnumeratorConstructorArgsTypes() {
            return new Class[0];
        }

        @Override
        public Object[] getEnumeratorConstructorArgs() {
            return new Object[0];
        }

        // Using org.webrtc.FlashlightCameraEnumerator to access flashlight
hidden controls.
        @Override
        public String getEnumeratorClassName() {
            return CustomCameras.FLASHLIGHT_CAMERA_ENUMERATOR;
        }
    };
}

```

2. Выбор камеры

[code](#)

```

CameraCapturerFactory.getInstance().setCustomCameraCapturerOptions(createCustomCameraCapturerOptions());

mCameraCapturer = (LabelledSpinner) findViewById(R.id.camera_capturer);
mCameraCapturer.setOnItemSelectedListener(new
LabelledSpinner.OnItemSelectedListener() {
    @Override
    public void onItemSelected(View labelledSpinner, AdapterView<?>
adapterView, View itemView, int position, long id) {
        String captureType =
getResources().getStringArray(R.array.camera_capturer)[position];
        switch (captureType) {
            case "flashlight":

CameraCapturerFactory.getInstance().setCameraType(CameraCapturerFactory.CameraType.FLASHLIGHT);

                break;
            case "camera1capturer":

CameraCapturerFactory.getInstance().setCameraType(CameraCapturerFactory.CameraType.FRONT_CAMERA);

                break;
            case "camera2capturer":

CameraCapturerFactory.getInstance().setCameraType(CameraCapturerFactory.CameraType.BACK_CAMERA);

                break;
            case "custom":

CameraCapturerFactory.getInstance().setCameraType(CameraCapturerFactory.CameraType.CUSTOM);

                break;
        }

mCameraSpinner.setItemsArray(Flashphoner.getMediaDevices().getVideoList());

    }

    @Override
    public void onNothingChosen(View labelledSpinner, AdapterView<?>
adapterView) {

    }

});

```