

항목 선택 (2)

Mobile Software
2022 Fall

All rights reserved, 2022 Copyright by Youn-Sik Hong (편집, 배포 불허)

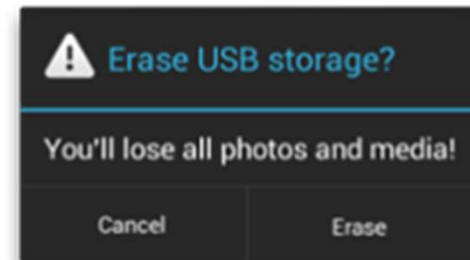
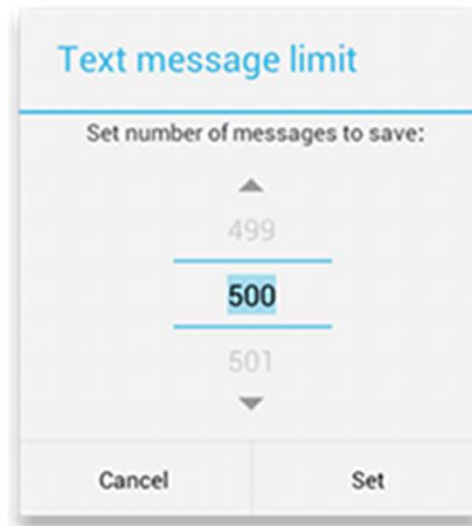
What to do next?

- 항목 선택
 - RadioButton
 - CheckBox
 - Spinner
 - **AlertDialog**

- 강의 노트에 포함된 코드 제공(7-3.소스코드. hwp)

Dialog (대화 상자)

- 사용자에게 메시지를 전달하거나, 사용자로부터 선택적 입력을 받기 위한 pop-up 창



- 종류
 - **AlertDialog**
 - **ProgressDialog**
 - **DatePickerDialog**
 - **TimePickerDialog**

AlertDialog

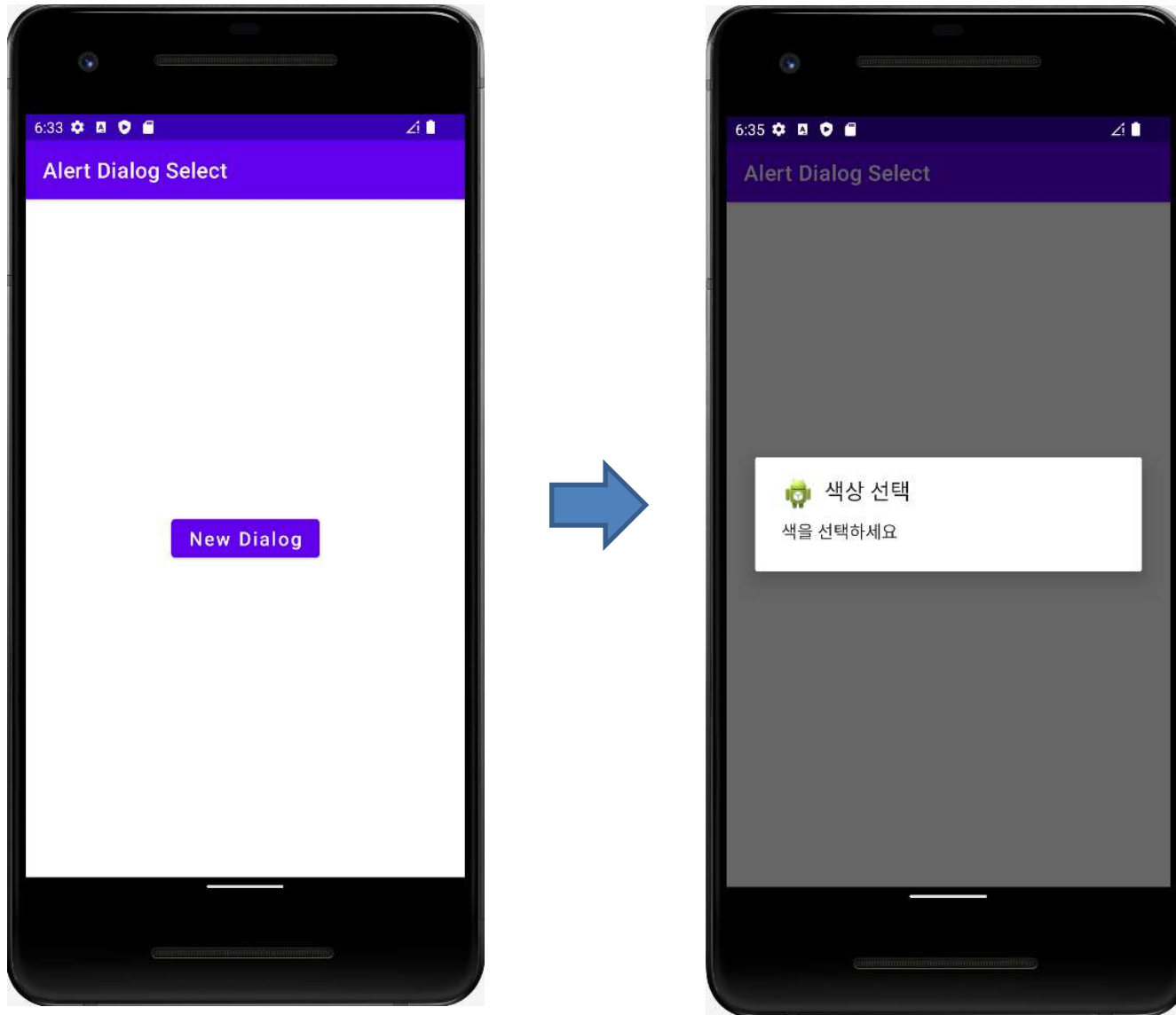
- 대화 상자의 기본 클래스는 Dialog 이지만,
 - 사용하기 쉽게 wrapping된 **AlertDialog** 클래스를 주로 사용
- AlertDialog 객체 생성: 내부 클래스인 **Builder** 사용
 - val dialog = AlertDialog.Builder (this@MainActivity)**
 - **this** 는 대화 상자를 생성하는 부모 activity를 가리킴
 - **AlertDialog.Builder** 에서 제공하는 메소드
 - **setMessage, setTitle, setIcon**
 - **create**
 - **show**

실습 프로젝트 생성

- 새 프로젝트 생성
 - Project name : **AlertDialog Select**
 - Package name : **com.example.alertdialogselect**
 - Activity : **Empty Activity**
 - Activity name : **MainActivity.kt**
 - Layout name : **activity_main.xml**
- 자동 생성된 XML 파일의 root view는 **ConstraintLayout**
 - TextView 삭제 → **Button** 을 중앙에 배치

ConvertView를 사용해서 Button 으로 변환
(recommended)

Basic AlertDialog



실습 1: A Basic AlertDialog

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        val callButton = findViewById<Button>(R.id.callButton)  
        callButton.setOnClickListener { it: View {  
            val builder = AlertDialog.Builder(context this@MainActivity)  
            builder.setTitle("색상 선택")  
            builder.setMessage("색을 선택하세요")  
            builder.setIcon(R.drawable.ic_launcher)  
            builder.show()  
        }  
    }  
}
```

applicationContext 를 사용하면
runtime error 발생

applicationContext는 앱이 실행될 때부터 끝날 때까지
유지됩니다. 그런데 dialog는 잠깐 보여졌다 닫히기 때문에
applicationContext를 호출하면, dialog 창이 닫힐 때 함께
applicationContext를 없애면 앱 실행이 중단됩니다!!!



잠깐! A Different Coding Style

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        val callButton = findViewById<Button>(R.id.callButton)  
        callButton.setOnClickListener { it: View!  
            val builder = AlertDialog.Builder(context: this@MainActivity)  
            builder.setTitle("색상 선택")  
            builder.setMessage("색을 선택하세요")  
            builder.setIcon(R.drawable.ic_launcher)  
            builder.show()  
        }  
    }  
}
```

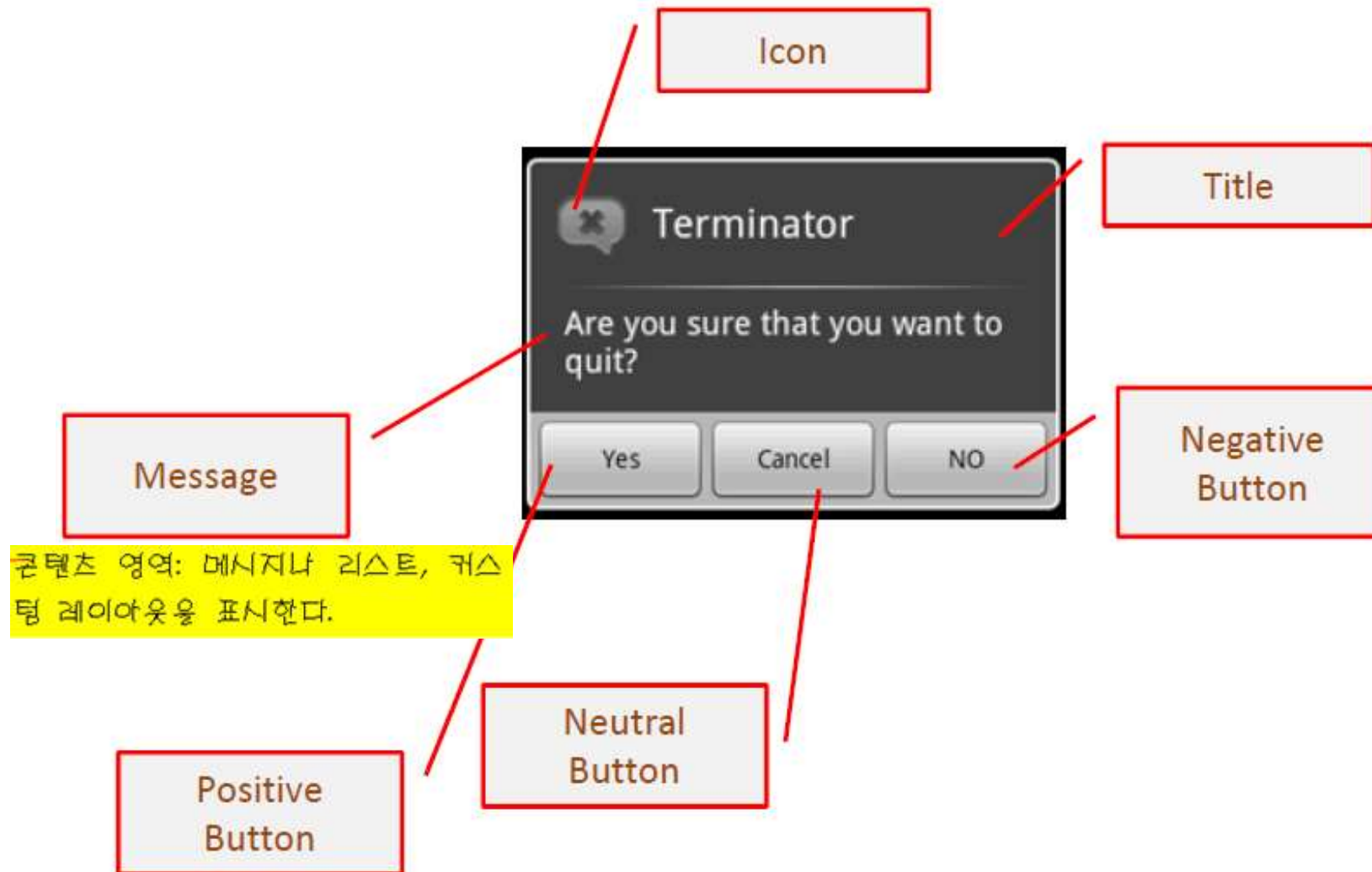
AlertDialog를 보여주기 위한
show() 메소드 호출



```
callButton.setOnClickListener { it: View!  
    AlertDialog.Builder(context: this@MainActivity)  
        .setTitle("색상 선택")  
        .setMessage("색을 선택하세요")  
        .setIcon(R.drawable.ic_launcher)  
        .show()  
}
```

```
callButton.setOnClickListener { it: View!  
    AlertDialog.Builder(context: this@MainActivity).apply { this: AlertDialog.Builder  
        setTitle("색을 선택하세요")  
        setIcon(R.drawable.ic_launcher)  
        setPositiveButton(text: "Select", listener: null)  
        setNegativeButton(text: "Cancel", listener: null)  
        show()  
    }  
}
```


A Complete AlertDialog




실습 2: 응답 버튼 추가

res

- drawable
 - ic_launcher.png
 - ic_launcher_background.xml
 - ic_launcher_foreground.xml (v24)

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        val callButton: Button = findViewById(R.id.callButton)  
        callButton.setOnClickListener { it: View!  
            AlertDialog.Builder(context: this@MainActivity)  
                .setTitle("색을 선택하세요")  
                .setIcon(R.drawable.ic_launcher)  
                .setPositiveButton(text: "Select", listener: null)  
                .setNegativeButton(text: "Cancel", listener: null)  
                .show()  
        }  
    }  
}
```

2번째 인자는 이벤트 핸들러

 색을 선택하세요
CANCEL SELECT

실습 3: 이벤트 핸들러 추가

```
callButton.setOnClickListener { it View!  
    AlertDialog.Builder( context: this@MainActivity)  
        .setTitle("색을 선택하세요")  
        .setIcon(R.drawable.ic_launcher)  
        .setPositiveButton( text: "Select", object: DialogInterface.OnClickListener {  
            override fun onClick(dialog: DialogInterface?, which: Int) {  
                finish()  
            }  
        })  
        .setNegativeButton( text: "Cancel", object: DialogInterface.OnClickListener {  
            override fun onClick(dialog: DialogInterface?, which: Int) {  
                Toast.makeText(  
                    applicationContext, text: "The command is cancelled.",  
                    Toast.LENGTH_SHORT  
                ).show()  
            }  
        })  
        .show()  
    }  
}
```

이벤트 핸들러
구현

App. 종료

문자열 배열 리소스

res/values/strings.xml

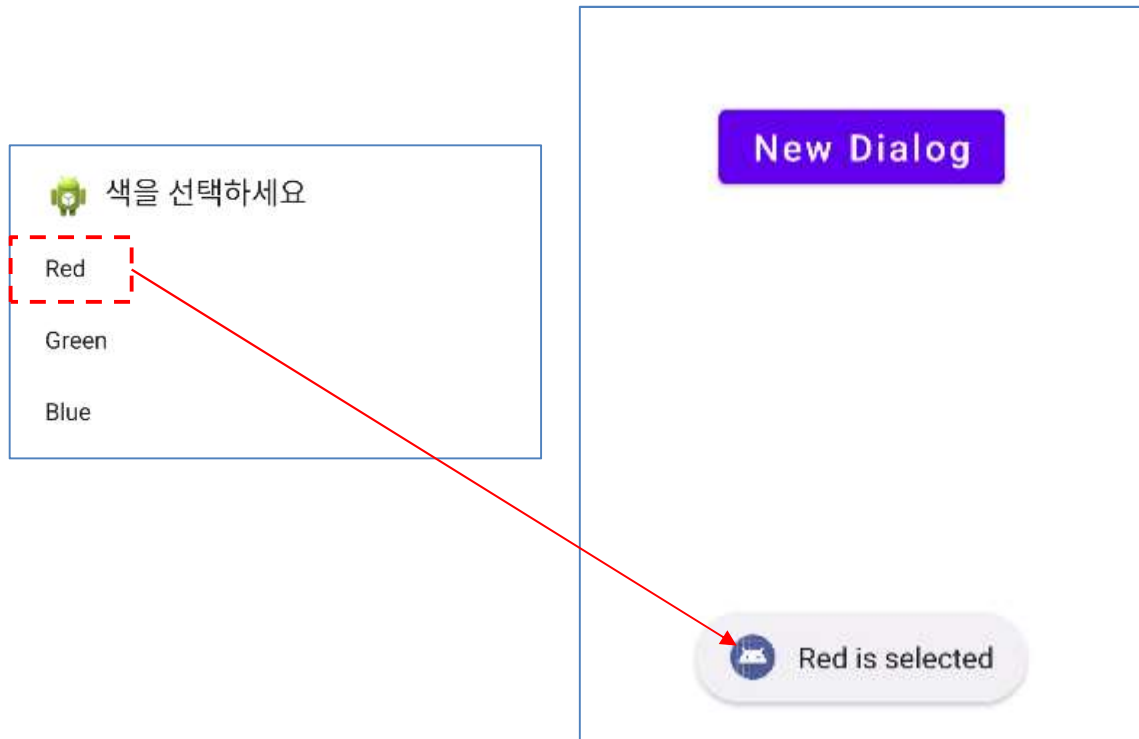
```
<resources>
  <string name="app_name">Alert Dialog Select</string>
  <string-array name="colors">
    <item>Red</item>
    <item>Green</item>
    <item>Blue</item>
  </string-array>
</resources>
```

실습 4: setItems()를 사용한 단일 항목 선택

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        val items:Array<String> = resources.getStringArray(R.array.colors)  
  
        val callButton:Button = findViewById(R.id.callButton)  
        callButton.setOnClickListener { it: View!  
            AlertDialog.Builder( context: this@MainActivity)  
                .setTitle("색을 선택하세요")  
                .setIcon(R.drawable.ic_launcher)  
                .setItems(items, object:DialogInterface.OnClickListener {  
                    override fun onClick(dialog: DialogInterface?, which: Int) {  
                        Toast.makeText(applicationContext,  
                            text: "${items[which]} is selected",  
                            Toast.LENGTH_SHORT).show()  
                    }  
                })  
            .show()  
        }  
    }  
}
```

which는
선택한 항목의
인덱스

실행 결과 : 실습 4



실습 5: 단일 항목 선택 - SingleChoiceItems

```
val items:Array<String> = resources.getStringArray(R.array.colors)
val callButton:Button = findViewById(R.id.callButton)
var mSelect = 0
```

```
callButton.setOnClickListener { it: View!
    AlertDialog.Builder(this@MainActivity)
        .setTitle("색을 선택하세요")
        .setIcon(R.drawable.ic_launcher)
        .setSingleChoiceItems(items, mSelect, object:DialogInterface.OnClickListener {
            override fun onClick(dialog: DialogInterface?, which: Int) {
                mSelect = which
            }
        })
        .setPositiveButton("Select", object: DialogInterface.OnClickListener{
            override fun onClick(dialog: DialogInterface?, which: Int) {
                Toast.makeText(applicationContext,
                    "${items[mSelect]} is selected",
                    Toast.LENGTH_SHORT).show()
            }
        })
        .setNegativeButton("Cancel", null)
        .show()
}
```



기본으로
선택된 항목.
mSelect = 0
(첫 번째 항목)

아무 것도
선택되지 않은
상태로 만들려면
mSelect = -1

mSelect =
선택한 항목

실습 5 – 람다 식 코드

```
val items:Array<String> = resources.getStringArray(R.array.colors)
var mSelect = 0

val callButton:Button = findViewById(R.id.callButton)
callButton.setOnClickListener { it View!
    AlertDialog.Builder( context this@MainActivity)
        .setTitle("색을 선택하세요")
        .setIcon(R.drawable.ic_launcher)
        .setSingleChoiceItems(items, mSelect) { _, which ->
            mSelect = which
        }
        .setPositiveButton( text: "Select") { _, _ ->
            Toast.makeText(
                applicationContext,
                text: "${items[mSelect]} is selected",
                Toast.LENGTH_SHORT
            ).show()
        }
        .setNegativeButton( text: "Cancel", listener: null)
        .show()
}
```


실습 6: 다중 항목 선택 – MultiChoiceItems

```
val items:Array<String> = resources.getStringArray(R.array.colors)
var checked:Array<Boolean> = Array( size: 3){false}
val callButton:Button = findViewById(R.id.callButton)
callButton.setOnClickListener { it: View!
    AlertDialog.Builder( context: this@MainActivity)
        .setTitle("색을 선택하세요")
        .setIcon(R.drawable.ic_launcher)
        .setMultiChoiceItems(items, checkedItems: null,
            object: DialogInterface.OnMultiChoiceClickListener {
                override fun onClick(dialog: DialogInterface?,
                    which: Int, isChecked: Boolean) {
                    checked[which] = isChecked
                }
            })
        .setPositiveButton( text: "Select",
            object:DialogInterface.OnClickListener{
                override fun onClick(dialog: DialogInterface?, which: Int) {
                    var sb = StringBuffer()
                    for (i in checked.indices) {
                        if (checked[i])
                            sb.append("${items[i]}, ")
                    }
                    Toast.makeText(applicationContext,
                        text: "$sb are selected", Toast.LENGTH_SHORT).show()
                }
            })
        .setNegativeButton( text: "Cancel", listener: null)
        .show()
}
```

항목 선택 정보를
저장하기 위한 배열

선택된 항목
업데이트

선택된 항목
추출

색을 선택하세요

- ☐ Red
- ☒ Green
- ☒ Blue

CANCEL SELECT

실습 6 – 람다 식 코드

```
val items:Array<String> = resources.getStringArray(R.array.colors)
var checked:Array<Boolean> = Array( size: 3) {false}

val callButton:Button = findViewById(R.id.callButton)
callButton.setOnClickListener { it: View!
    AlertDialog.Builder( context: this@MainActivity)
        .setTitle("색을 선택하세요")
        .setIcon(R.drawable.ic_launcher)
        .setMultiChoiceItems(items, checkeditems: null) { _, which, isChecked ->
            checked[which] = isChecked
        }
        .setPositiveButton( text: "Select") { _, _ ->
            var sb = StringBuffer()
            for (i in checked.indices) {
                if (checked[i])
                    sb.append("${items[i]}, ")
            }
            Toast.makeText(applicationContext,
                text: "$sb are selected", Toast.LENGTH_SHORT).show()
        }
        .setNegativeButton( text: "Cancel", listener: null)
        .show()
}
```

기본으로
선택된 항목
(없음)