# ЛАБОРАТОРНА РОБОТА №3

# БАГАТОПОТОЧНІСТЬ В ОС ANDROID

**Мета роботи:**

* Ознайомитись з основними принципами роботи з асинхронними подіями в ОС Android.

## Хід виконання роботи

### Завдання згідно з варіантом №8

Модифікувати розроблений у попередній лабораторній роботі додаток таким чином, щоб налаштування завантажувались з Shared Preferences за допомогою ExecutorService.

### Результати виконання зі скріншотом екрану результатів розрахунку варіанту завдання

Було змінено код класу GameActivity його наведено в лістингу 3.1 – це клас який відображає за допомогою layout activity\_game екран гри.

Лістинг 3.1 – GameActivity

package com.hellstaff.repeattheword  
  
import android.annotation.SuppressLint  
import android.app.AlertDialog  
import android.content.SharedPreferences  
import android.os.Bundle  
import android.os.CountDownTimer  
import android.util.Log  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import androidx.preference.PreferenceManager  
import androidx.recyclerview.widget.GridLayoutManager  
import com.hellstaff.repeattheword.databinding.ActivityGameBinding  
import java.util.concurrent.Executors  
  
class GameActivity : AppCompatActivity(), Adaptor.CheckWord {  
 private lateinit var binding: ActivityGameBinding  
 private var length: Int = 0  
 private var time: Long = 0  
 private var counterTrueWord: Short = 0  
 private var timerWork: Boolean = false  
 private var timer: CountDownTimer? = null  
 private var word: String = ""  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 binding = ActivityGameBinding.inflate(layoutInflater)  
 setContentView(binding.root)  
 val sharedPreferences = PreferenceManager.getDefaultSharedPreferences(this)  
 val service = Executors.newScheduledThreadPool(2)  
 service.execute**{** length = sharedPreferences.getString("word", "4")!!.toInt()  
 time = sharedPreferences.getString("timer", "60000")!!.toLong()**}** service.shutdown()  
 }  
  
 override fun onStart() {  
 super.onStart()  
 counterTrueWord = 0  
 startCountDownTimer(time)  
 timerWork = true  
 }  
  
 override fun onResume() {  
 super.onResume()  
 val letters = fetchDate()  
 letters.shuffle()  
 val adaptor = Adaptor(this, letters, fetchDate("-"))  
 binding.word.layoutManager = GridLayoutManager(this, length)  
 binding.word.adapter = adaptor  
 }  
  
 private fun fetchDate(): ArrayList<String> {  
 var letters = ""  
 val wordTrue = ArrayList<String>()  
 when (length) {  
 4 -> {  
 val number = (0..resources.getStringArray(R.array.word\_4\_char).size).random()  
 letters = resources.getStringArray(R.array.word\_4\_char)[number]  
 }  
 5 -> {  
 val number = (0..resources.getStringArray(R.array.word\_5\_char).size).random()  
 letters = resources.getStringArray(R.array.word\_5\_char)[number]  
 }  
 6 -> {  
 val number = (0..resources.getStringArray(R.array.word\_6\_char).size).random()  
 letters = resources.getStringArray(R.array.word\_6\_char)[number]  
 }  
 7 -> {  
 val number = (0..resources.getStringArray(R.array.word\_7\_char).size).random()  
 letters = resources.getStringArray(R.array.word\_7\_char)[number]  
 }  
 8 -> {  
 val number = (0..resources.getStringArray(R.array.word\_8\_char).size).random()  
 letters = resources.getStringArray(R.array.word\_8\_char)[number]  
 }  
 9 -> {  
 val number = (0..resources.getStringArray(R.array.word\_9\_char).size).random()  
 letters = resources.getStringArray(R.array.word\_9\_char)[number]  
 }  
 }  
 word = letters  
 letters.forEach **{** element **->** wordTrue.add(element.toString()) **}** return wordTrue  
 }  
  
 private fun fetchDate(symbol: String): ArrayList<String> {  
 val item = ArrayList<String>()  
 for (i in 0 until length) {  
 item.add(symbol)  
 }  
 return item  
 }  
  
 private fun startCountDownTimer(timeMillis: Long) {  
 timer?.cancel()  
 timer = object : CountDownTimer(timeMillis, 1000) {  
 @SuppressLint("SetTextI18n")  
 override fun onTick(timeMili: Long) {  
 if ((timeMili / 1000 % 60).toString().length == 1) binding.textTimer.text =  
 "0" + (timeMili / 60000).toString() + ":0" + (timeMili / 1000 % 60).toString()  
 else binding.textTimer.text =  
 "0" + (timeMili / 60000).toString() + ":" + (timeMili / 1000 % 60).toString()  
 }  
  
 override fun onFinish() {  
 timerWork = false  
 binding.textTimer.text = "Час вийшов!"  
 onCreateDialog()  
 }  
 }.start()  
 }  
  
 override fun checkWord(word: ArrayList<String>) {  
 Log.d("checkWord", word.joinToString(""))  
 if (word.joinToString("") == this.word) {  
 counterTrueWord++  
 Toast.makeText(this, "Вгадано! $counterTrueWord", Toast.LENGTH\_LONG).show()  
 onResume()  
 }  
 }  
  
 private fun onCreateDialog() {  
 val builder = AlertDialog.Builder(this)  
 builder.setTitle("Гру завершено")  
 builder.setMessage("Ваш результат: $counterTrueWord")  
 builder.setNeutralButton("Почати ще раз") **{** \_, \_ **->** onStart() **}** builder.show()  
 }  
}

## Висновки

На даній лабораторній роботі вдалося ознайомитись з основними принципами роботи з асинхронними подіями в ОС Android.