Analysis of Synchronize Singleton

(Android Example)

# Case1: No Synchronized

//src

|  |
| --- |
| public static SingletonP1 getInstance() {  if (sInstance == null) {  sInstance = new SingletonP1();  }  return sInstance;  } |

// dex

|  |
| --- |
| |[000110] com.adam.app.SingletonP1.getInstance:()Lcom/adam/app/SingletonP1;  |0000: sget-object v0, Lcom/adam/app/SingletonP1;.sInstance:Lcom/adam/app/SingletonP1; // field@0000  |0002: if-nez v0, 000b // +0009  |0004: new-instance v0, Lcom/adam/app/SingletonP1; // type@0000  |0006: invoke-direct {v0}, Lcom/adam/app/SingletonP1;.<init>:()V // method@0000  |0009: sput-object v0, Lcom/adam/app/SingletonP1;.sInstance:Lcom/adam/app/SingletonP1; // field@0000  |000b: sget-object v0, Lcom/adam/app/SingletonP1;.sInstance:Lcom/adam/app/SingletonP1; // field@0000  |000d: return-object v0 |

# Case2: Synchronized

//src

|  |
| --- |
| public static synchronized SingletonP2 getInstance() {  if (sInstance == null) {  sInstance = new SingletonP2();  }  return sInstance;  } |

//dex

|  |
| --- |
| |[000110] com.adam.app.SingletonP2.getInstance:()Lcom/adam/app/SingletonP2;  |0000: const-class v1, Lcom/adam/app/SingletonP2; // type@0000  |0002: monitor-enter v1  |0003: sget-object v0, Lcom/adam/app/SingletonP2;.sInstance:Lcom/adam/app/SingletonP2; // field@0000  |0005: if-nez v0, 000e // +0009  |0007: new-instance v0, Lcom/adam/app/SingletonP2; // type@0000  |0009: invoke-direct {v0}, Lcom/adam/app/SingletonP2;.<init>:()V // method@0000  |000c: sput-object v0, Lcom/adam/app/SingletonP2;.sInstance:Lcom/adam/app/SingletonP2; // field@0000  |000e: sget-object v0, Lcom/adam/app/SingletonP2;.sInstance:Lcom/adam/app/SingletonP2; // field@0000  |0010: monitor-exit v1  |0011: return-object v0  |0012: move-exception v0  |0013: monitor-exit v1  |0014: throw v0 |

# Case3: Add volatile keyword in synchronized instance

//src

|  |
| --- |
| private static volatile SingletonP3 sInstance;    private SingletonP3() {}    public static synchronized SingletonP3 getInstance() {  if (sInstance == null) {  sInstance = new SingletonP3();  }  return sInstance;  } |

//dex

|  |
| --- |
| Static fields -  #0 : (in Lcom/adam/app/SingletonP3;)  name : 'sInstance'  type : 'Lcom/adam/app/SingletonP3;'  access : 0x004a (PRIVATE STATIC VOLATILE) |

the flow of getInstance is the same as the case2.

# Case4: Refactor synchronized getinstance method as using synchronized block

//src

|  |
| --- |
| private static volatile SingletonP4 sInstance;  private SingletonP4() {}    public static SingletonP4 getInstance() {  synchronized(SingletonP4.class) {  if (sInstance == null) {  sInstance = new SingletonP4();  }  }  return sInstance;  } |

//dex

|  |
| --- |
| |[000120] com.adam.app.SingletonP4.getInstance:()Lcom/adam/app/SingletonP4;  |0000: const-class v1, Lcom/adam/app/SingletonP4; // type@0000  |0002: monitor-enter v1  |0003: sget-object v0, Lcom/adam/app/SingletonP4;.sInstance:Lcom/adam/app/SingletonP4; // field@0000  |0005: if-nez v0, 000e // +0009  |0007: new-instance v0, Lcom/adam/app/SingletonP4; // type@0000  |0009: invoke-direct {v0}, Lcom/adam/app/SingletonP4;.<init>:()V // method@0000  |000c: sput-object v0, Lcom/adam/app/SingletonP4;.sInstance:Lcom/adam/app/SingletonP4; // field@0000  |000e: monitor-exit v1  |000f: sget-object v0, Lcom/adam/app/SingletonP4;.sInstance:Lcom/adam/app/SingletonP4; // field@0000  |0011: return-object v0  |0012: move-exception v0  |0013: monitor-exit v1  |0014: throw v0 |

# Case5: Double check lock pattern

//src

|  |
| --- |
| private static volatile SingletonP5 sInstance;    private SingletonP5() {}    public static SingletonP5 getInstance() {  if (sInstance == null) {  synchronized (SingletonP5.class) {  if (sInstance == null) {  sInstance = new SingletonP5();  }  }  }  return sInstance;  } |

//dex

|  |
| --- |
| |[000120] com.adam.app.SingletonP5.getInstance:()Lcom/adam/app/SingletonP5;  |0000: sget-object v0, Lcom/adam/app/SingletonP5;.sInstance:Lcom/adam/app/SingletonP5; // field@0000  |0002: if-nez v0, 0013 // +0011  |0004: const-class v1, Lcom/adam/app/SingletonP5; // type@0000  |0006: monitor-enter v1  |0007: sget-object v0, Lcom/adam/app/SingletonP5;.sInstance:Lcom/adam/app/SingletonP5; // field@0000  |0009: if-nez v0, 0012 // +0009  |000b: new-instance v0, Lcom/adam/app/SingletonP5; // type@0000  |000d: invoke-direct {v0}, Lcom/adam/app/SingletonP5;.<init>:()V // method@0000  |0010: sput-object v0, Lcom/adam/app/SingletonP5;.sInstance:Lcom/adam/app/SingletonP5; // field@0000  |0012: monitor-exit v1  |0013: sget-object v0, Lcom/adam/app/SingletonP5;.sInstance:Lcom/adam/app/SingletonP5; // field@0000  |0015: return-object v0  |0016: move-exception v0  |0017: monitor-exit v1  |0018: throw v0 |

# Case6: Use enum pattern to do synchronized singleton

//src

|  |
| --- |
| public enum SingletonP6 {  INSTANCE;    private SingletonP6() {}    } |

//dex

|  |
| --- |
| |[0001c0] com.adam.app.SingletonP6.<clinit>:()V  |0000: const/4 v2, #int 0 // #0  |0001: new-instance v0, Lcom/adam/app/SingletonP6; // type@0001  |0003: const-string v1, "INSTANCE" // string@0006  |0005: invoke-direct {v0, v1, v2}, Lcom/adam/app/SingletonP6;.<init>:(Ljava/lang/String;I)V // method@0001  |0008: sput-object v0, Lcom/adam/app/SingletonP6;.INSTANCE:Lcom/adam/app/SingletonP6; // field@0001  |000a: const/4 v0, #int 1 // #1  |000b: new-array v0, v0, [Lcom/adam/app/SingletonP6; // type@0009  |000d: sget-object v1, Lcom/adam/app/SingletonP6;.INSTANCE:Lcom/adam/app/SingletonP6; // field@0001  |000f: aput-object v1, v0, v2  |0011: sput-object v0, Lcom/adam/app/SingletonP6;.ENUM$VALUES:[Lcom/adam/app/SingletonP6; // field@0000  |0013: return-void |

Above this flow is in the class initialization process(<clinit>)

Ps: **By default, the Enum instance is thread-safe, and you don’t need to worry about double-checked locking.**

# Convert to dex info from class

1. Put class file and toDexDumpFile.bat in as the flowing directory

<Android sdk>\build-tools\<version>

Ps: The toDexDumpFile.bat file is put the tool folder.