

# Mobile\_logs

## Уровни логирования

Логи - это текстовые файлы, в которых записываются все действия пользователя. Например, какие кнопки он нажимает в приложении и как на это оно реагирует в ответ.

Записи в логах формируются в хронологическом порядке. Самая свежая — внизу.

Есть два вида логов:

- **Crash logs** — файл, в котором хранятся записи только об ошибках экстренного завершения программы — по-простому, когда приложение крашнулось.

- **Logs** — простые логи, или журнал событий. Это файл, в котором хранятся системные записи и ответы устройства на действие пользователя.

Логи на мобильных устройствах бывают нескольких уровней:

- ERROR,
- WARN,
- INFO,
- DEBUG,
- VERBOSE.

Они представлены по уровню важности — от самого высокого к самому низкому, — и каждый следующий уровень включает в себя предыдущий. Например, VERBOSE содержит в себе логи всех остальных.

**Примечание:** уровни более применимы к логам на Android, потому что именно там такое разделение встречается чаще.

Рассмотрим подробнее каждый уровень.

### Error (ERROR)

На этом уровне информируются ошибки работы системы.

Записи этого уровня требуют быстрого вмешательства разработчика — на такие ошибки нужно реагировать максимально быстро.

Как пример, такая запись в логе:

```
“ SpannableStringBuilder: SPAN_EXCLUSIVE_EXCLUSIVE spans cannot have a zero length ”
```

Это ошибка, в которой говорится, что строковый элемент span не может быть пустым.

Или вот:

```
“ [ZeroHung]zrhung_get_config: Get config failed for wp[0x0008] ] ”
```

Эта системная ошибка сообщает, что происходит утечка памяти при взаимодействии с каким-то элементом или приложением.

### **Warning (WARN)**

На этом уровне отображаются записи, сообщающие о каком-то неожиданном поведении, требующем внимания, или о ситуации, которая незнакома системе.

Например, сообщение ниже — запись из тестового приложения:

```
“ [OMX.hisi.video.decoder.avc] setting nBufferCountActual to 16 failed: -2147483648 ”
```

Мы пытаемся декодировать запись в какой-то формат, но его нет. Ошибка сообщает о неуспешной попытке настройки видеоплеера в нужном формате.

Ещё пример:

```
“ BroadcastQueue: Permission Denial: broadcasting Intent ”
```

Эта системная ошибка говорит о сбое в работе одного из виджетов на устройстве.

### **Info (INFO)**

На этот уровень приходят записи информационного характера, например о работе системы.

Допустим, такое сообщение об уровне заряда батареи на устройстве:

```
“ APwBatteryMonitor: screen off start battery: 100 ”
```

А это сообщение говорит о том, что экран устройства был выключен:

```
“ HwBatteryService: intent = Intent { act=android.intent.action.SCREEN_OFF flg=0x58200010 } ”
```

Ещё в логи этого уровня входят запросы от клиента на сервер: хедеры, тело запросов, которые отправляет клиент, и ответы сервера.

```
“ okhttp.OkHttpClient: <-- 200 https://domainname/api/v1/smith/deals (1691ms)
```

```
okhttp.OkHttpClient: server: nginx/1.15.9
```

```
okhttp.OkHttpClient: date: Thu, 23 Sep 2021 19:41:17 GMT
```

```
okhttp.OkHttpClient: content-type: application/json
```

```
okhttp.OkHttpClient: vary: Accept-Encoding
```

```
okhttp.OkHttpClient:      strict-transport-security:      max-age=15724800;  
includeSubDomains
```

```
okhttp.OkHttpClient: { "key": { "key": value, "name": "" }, "key": value, "key": value }
```

```
okhttp.OkHttpClient: <-- END HTTP ”
```

Такие записи могут помочь вам в понимании какого-то бага или в разборе задачи при условии, что вы не можете перехватить трафик или не знаете, какие запросы отправляются на бэкенд.

## **Debug (DEBUG)**

Это уровень сообщений, в которых передаётся информация о процессах отладки или шагах работы крупных процессов.

Например, в записи ниже сказано, что пользователь нажимал на кнопку уменьшения или увеличения громкости:

```
“ MediaSessionService: dispatchVolumeKeyEvent ”
```

Сначала мы видим запись о самом факте нажатия на кнопку, далее оно расшифровывается подробнее:

```
{ action=ACTION_DOWN, keyCode=KEYCODE_VOLUME_UP }
```

Ещё пример: если ваше приложение использует сокет-сессию, то на уровне DEBUG мы можем увидеть, когда сессия начинается и заканчивается:

```
“ b$b: WebSocket connected ”
```

## **Verbose (VERBOSE)**

Сообщения такого уровня уточняют или раскрывают действия.

Например, у нас есть служба управления окнами на экране приложения. И на уровне Verbose мы можем увидеть подробности её работы.

Открытие окна:

WindowManager: addWindow

Заккрытие окна:

WindowManager: Removing Window

На этом уровне мы можем посмотреть системные подробности наших действий. Например, при включении геолокации в записи отобразится текущая геолокация.

GnssLocationProvider: reportLocation Location [...]

А меняя звук на устройстве, мы увидим, как растёт или падает значение:

AudioManager: getStreamVolume streamType: 3 volume: 10

Каждое нажатие, то есть изменение звука, будет отражаться новым сообщением.

Verbose — уровень самого низкого приоритета. Выбирая такой уровень отображения логов, мы будем видеть записи и со всех предыдущих уровней.

**Примечание:** разработчики приложения самостоятельно покрывают действия логами, определяют уровни, а также какие сообщения какому из них соответствуют.

## Инструменты для снятия логов

### Android

Расскажем о трёх способах.

Первый — Logcat в составе Android Studio, самый известный и широко используемый.

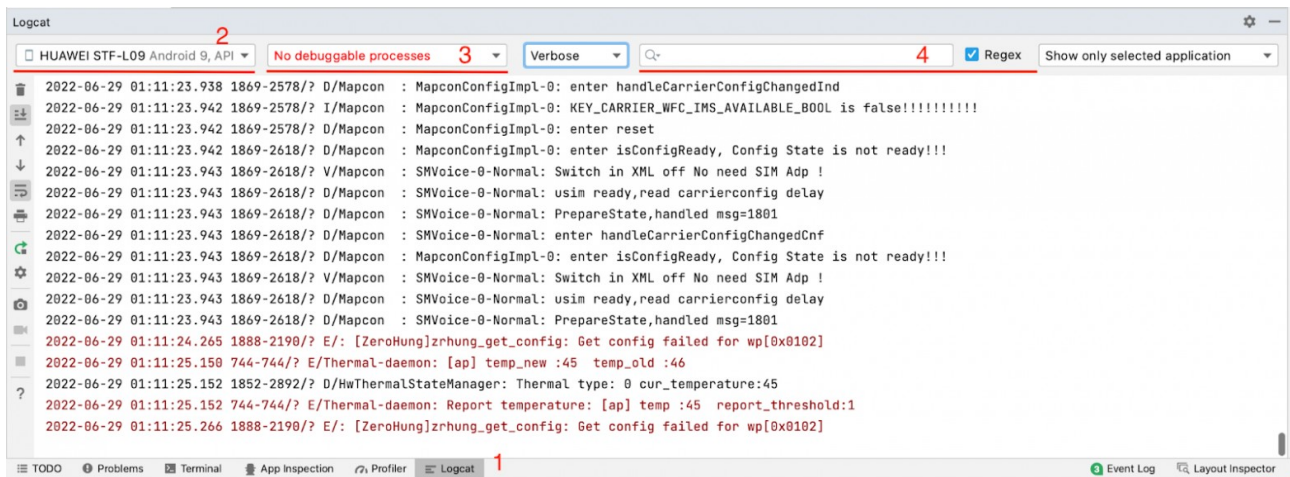
Для снятия логов нам необходимо перевести устройство в режим разработчика/отладки. Для этого нужно:

- найти в настройках номер нашего билда или ОС (в зависимости от устройства),
- около десяти раз нажать на эту информацию,
- при появлении сообщения о том, не хотим ли мы перевести устройство в режим разработчика, нажать «Ок».

Примечание: алгоритм может отличаться в зависимости от производителя устройства, потому что у многих из них свои надстройки на ОС Android.

Дальше подключаем устройство по USB к ПК и устанавливаем Android Studio.

Следующие шаги на скрине:



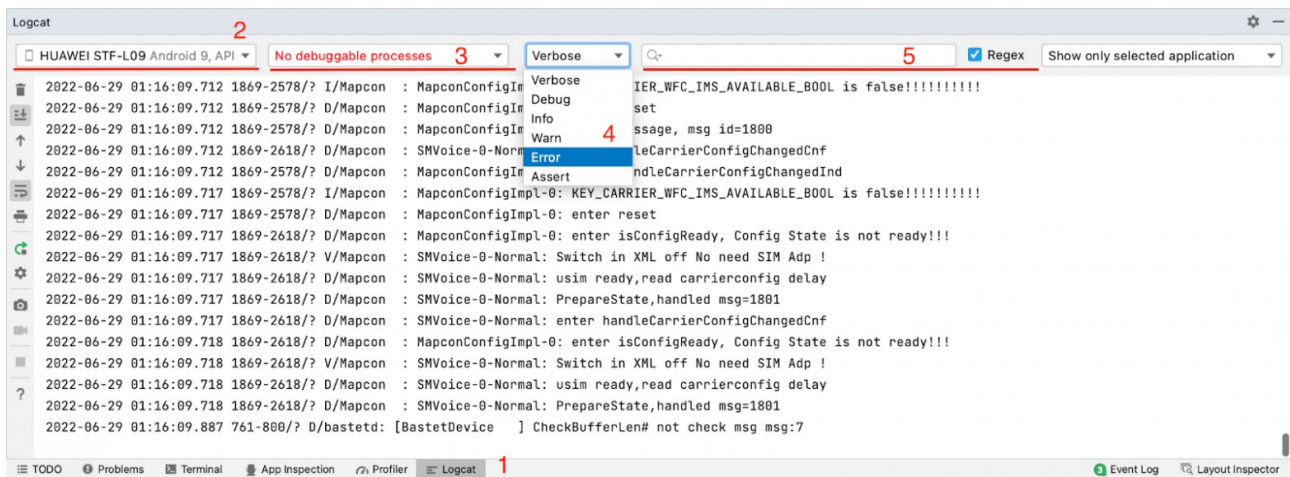
1.Выбираем вкладку Logcat (переходим к сообщениям в реальном времени).

2.В окошке выбираем телефон, с которого снимаем логи.

3.На этой вкладке выбираем логи определённого приложения. Если нужно снять вообще все логи со всех приложений и системы, эту вкладку стоит не трогать. Рядом с ней можно выбрать уровень логирования (вкладка Verbose на скрине).

4.В поле поиска, где мы можем фильтровать выдачу, разрешено писать что угодно — от названия пакета до частей вроде fatal.

На скрине видно логи с подключенного устройства.



Второй способ — выгрузка логов с самого устройства. Кроме режима разработчика нам нужно подключить устройство к ПК через USB и установить ADB — Android Debug Bridge.

Открываем терминал и пишем две команды.

Первая — `adb devices` — показывает подключённые устройства, которые видит ADB. В терминале выглядит так:

```
-zsh
Last login: Wed Jun 29 01:18:34 on ttys001
o.nikitina@onikitina-new ~ % adb devices
List of devices attached
7BKDU18504001505    device

o.nikitina@onikitina-new ~ % adb -s 7BKDU18504001505 logcat
```

Название устройства — 7BKDU18504001505

Вводим вторую команду — `adb -s название устройства logcat`, — которая запускает утилиту Logcat для конкретного устройства. В терминале в реальном времени будут поступать логи.

```
-zsh
06-29 01:19:59.845 1666 1666 D HwCustMobileSignalControllerImpl: updateDataType, mDataNetType: 19, isCAstate: true
06-29 01:19:59.848 1326 1746 E : [ZeroHung]zrhung_get_config: Get config failed for wp[0x0000]
06-29 01:20:00.006 1326 1326 V AlarmManager: Received TIME_TICK alarm; rescheduling
06-29 01:20:00.008 1326 1326 I HwAlarmManagerService: hwSetAlarm listenerTag: time_tick
06-29 01:20:00.021 1666 1666 W PanelView: set notification panel padding = 1638
06-29 01:20:00.021 1666 1666 W HwBackDropView: setAnimationParamInner 0.0 0
06-29 01:20:00.026 1666 1666 W BokehDrawable: drawScrim colorB: 0 0
06-29 01:20:00.030 1666 1666 E DateView: DateView,mCurrentTime: 1656454800030
06-29 01:20:00.033 1666 1666 I EventCenter: EventCenter Get :android.intent.action.TIME_TICK
06-29 01:20:00.034 1666 1666 W ClockView1: action android.intent.action.TIME_TICK
06-29 01:20:00.034 1666 1666 I chatty : uid=10039(com.huawei.HwMultiScreenShot) com.android.systemui identical 1 line
06-29 01:20:00.034 1666 1666 W ClockView1: action android.intent.action.TIME_TICK
06-29 01:20:00.039 1666 1666 W PanelView: set notification panel padding = 1638
06-29 01:20:00.040 1666 1666 W HwBackDropView: setAnimationParamInner 0.0 0
06-29 01:20:00.045 1666 1666 W BokehDrawable: drawScrim colorB: 0 0
06-29 01:20:00.050 1666 1666 I LocalCalendar: CalendarId: 0000LocalCalendar Off
06-29 01:20:00.054 1666 1666 I chatty : uid=10039(com.huawei.HwMultiScreenShot) com.android.systemui identical 1 line
06-29 01:20:00.058 1666 1666 I LocalCalendar: CalendarId: 0000LocalCalendar Off
06-29 01:20:00.064 1666 1666 W PanelView: set notification panel padding = 1638
06-29 01:20:00.064 1666 1666 W HwBackDropView: setAnimationParamInner 0.0 0
06-29 01:20:00.065 1888 1888 I HwLauncher: Model onReceive intent=Intent { act=android.intent.action.TIME_TICK flg=0x50200014 hwFlg=0x900 (has extras) }
06-29 01:20:00.065 1888 1888 I HwLauncher: Model onReceive user=UserHandle{0}
06-29 01:20:00.068 1666 1666 W BokehDrawable: drawScrim colorB: 0 0
06-29 01:20:00.074 1666 1922 I KeyguardStatusView: Runnable for refresh
06-29 01:20:00.102 1326 9458 V BroadcastQueue: Finished with ordered broadcast BroadcastRecord{ef08532 u-1 android.intent.action.TIME_TICK}
```

Как их читать?

1. В первом столбце — дата и время поступления записи.
2. Во втором — обозначения уровней логирования. Например, D — это Debug.
3. В третьем показываются названия инструмента, утилиты, пакета, от которых поступает сообщение, а также расшифровка того, что вообще происходит.

Третий инструмент — SDK Platform Tools. Процесс его установки практически аналогичен предыдущим двум:

- переводим телефон в режим разработчика,
- подключаем к ПК по USB,
- скачиваем на ПК папку SDK PT (под свою ОС),
- открываем папку SDK PT в терминале.

Теперь пишем команду `./adb logcat -v threadtime > ./android-debug.log`.

В терминале это выглядит так:



```
-zsh
Last login: Wed Jun 29 01:18:34 on ttys001
o.nikitina@onikitina-new ~ % adb devices
List of devices attached
7BKDU18504001505      device

o.nikitina@onikitina-new ~ % adb -s 7BKDU18504001505 logcat
```

Прерываем выполнение команды (например, на Мас это Control+C). Лог добавляется в папку.

Имя	Дата изменения	Размер	Тип
android-debug.log	Сегодня, 01:26	1,2 МБ	Файл журнала
.log.txt	16 нояб. 2021 г., 21:15	1,4 МБ	Простой текст
.log.log	16 нояб. 2021 г., 21:13	817 КБ	Файл журнала
adb	20 июля 2021 г., 00:21	7,3 МБ	Испол...йл Unix
mke2fs	20 июля 2021 г., 00:21	798 КБ	Испол...йл Unix
sqlite3	20 июля 2021 г., 00:21	1,3 МБ	Испол...йл Unix
e2fsdroid	20 июля 2021 г., 00:21	1,5 МБ	Испол...йл Unix

Открываем:

```
android-debug.log
Показать Сейчас Очистить Перезагрузить Поделиться
06-29 01:26:14.710 1852 2970 I PGServer: getIopFrontApp. calling pkg: android
06-29 01:26:14.715 751 2964 I AwareLog: HibStrategy::HibStrategySwapCandidateProcessRemove packagename is com.huawei.android.hwouc
06-29 01:26:14.715 751 2964 I AwareLog: HiberManagerService :sendMessageToHiberTask successful
06-29 01:26:14.852 1815 1815 E CHR_ChrModemStatistics: sensors invalid
06-29 01:26:15.180 1326 7092 D SmartHeartBeat: isPendingPackage, false, pkg:com.android.deskclock, action:com.android.deskclock.ALARM_ALERT,not
pending alarm with AlarmClockInfo
06-29 01:26:15.235 1326 3372 D SmartHeartBeat: isPendingPackage, false, pkg:com.android.deskclock, action:com.android.deskclock.ALARM_ALERT,not
pending alarm with AlarmClockInfo
06-29 01:26:15.561 12408 12418 E : [ZeroHung]zrhung_get_config: Get config failed for wp[0x0008]
06-29 01:26:15.572 1888 2190 E : [ZeroHung]zrhung_get_config: Get config failed for wp[0x0102]
06-29 01:26:15.789 12408 12408 D AwareBitmapCacher: handleInit switch not opened pid=12408
06-29 01:26:16.110 744 744 E Thermal-daemon: [charger_ic] temp_new:47 temp_old:46
06-29 01:26:16.112 1852 2892 D HwThermalStateManager: Thermal type: 2 cur_temperature:47
06-29 01:26:16.112 744 744 E Thermal-daemon: Report temperature: [charger_ic] temp:47 report_threshold:1
06-29 01:26:16.114 744 744 E Thermal-daemon: [apl] temp_new:46 temp_old:45
06-29 01:26:16.116 1852 2892 D HwThermalStateManager: Thermal type: 0 cur_temperature:46
06-29 01:26:16.116 744 744 E Thermal-daemon: Report temperature: [apl] temp:46 report_threshold:1
06-29 01:26:16.517 1869 2618 D Mapcon : SMVoice-0-Normal: enter handleCarrierConfigChangedInd
06-29 01:26:16.518 1869 2618 D Mapcon : SMVoice-0-Normal: PrepareState,handled msg=CMD_CARRIER_CONFIG_CHANGE_IND
06-29 01:26:16.518 1869 2618 D Mapcon : SMVoice-0-Normal: enter handleCarrierConfigChangedInd
06-29 01:26:16.518 1869 2618 D Mapcon : SMVoice-0-Normal: PrepareState,handled msg=CMD_CARRIER_CONFIG_CHANGE_IND
06-29 01:26:16.520 1869 2578 D Mapcon : MapconConfigImpl-0: handleMessage, msg id=1800
06-29 01:26:16.520 1869 2578 D Mapcon : MapconConfigImpl-0: enter handleCarrierConfigChangedInd
06-29 01:26:16.526 1869 2578 I Mapcon : MapconConfigImpl-0: KEY_CARRIER_WFC_IMS_AVAILABLE_BOOL is false!!!!!!!!!!!!
06-29 01:26:16.526 1869 2578 D Mapcon : MapconConfigImpl-0: enter reset
06-29 01:26:16.526 1869 2578 D Mapcon : MapconConfigImpl-0: handleMessage, msg id=1800
06-29 01:26:16.526 1869 2618 D Mapcon : SMVoice-0-Normal: enter handleCarrierConfigChangedCnf
06-29 01:26:16.526 1869 2578 D Mapcon : MapconConfigImpl-0: enter handleCarrierConfigChangedInd
06-29 01:26:16.531 1869 2578 I Mapcon : MapconConfigImpl-0: KEY_CARRIER_WFC_IMS_AVAILABLE_BOOL is false!!!!!!!!!!!!
06-29 01:26:16.532 1869 2578 D Mapcon : MapconConfigImpl-0: enter reset
06-29 01:26:16.532 1869 2618 D Mapcon : MapconConfigImpl-0: enter isConfigReady, Config State is not ready!!!
06-29 01:26:16.532 1869 2618 V Mapcon : SMVoice-0-Normal: Switch in XML off No need SIM Adp !
06-29 01:26:16.532 1869 2618 D Mapcon : SMVoice-0-Normal: usim ready,read carrierconfig delay
06-29 01:26:16.532 1869 2618 D Mapcon : SMVoice-0-Normal: PrepareState,handled msg=1801
06-29 01:26:16.532 1869 2618 D Mapcon : SMVoice-0-Normal: enter handleCarrierConfigChangedCnf
06-29 01:26:16.532 1869 2618 D Mapcon : MapconConfigImpl-0: enter isConfigReady, Config State is not ready!!!
06-29 01:26:16.533 1869 2618 V Mapcon : SMVoice-0-Normal: Switch in XML off No need SIM Adp !
06-29 01:26:16.533 1869 2618 D Mapcon : SMVoice-0-Normal: usim ready,read carrierconfig delay
06-29 01:26:16.533 1869 2618 D Mapcon : SMVoice-0-Normal: PrepareState,handled msg=1801
06-29 01:26:16.573 1888 2190 E : [ZeroHung]zrhung_get_config: Get config failed for wp[0x0102]
```



В первом столбце — дата и время, во втором — уровни логов, в третьем — указание на то, от какой части системы поступают данные, лог и его расшифровка/подробности.

Очень похоже на предыдущий терминал, но файл обновляется, пока в терминале действует команда.

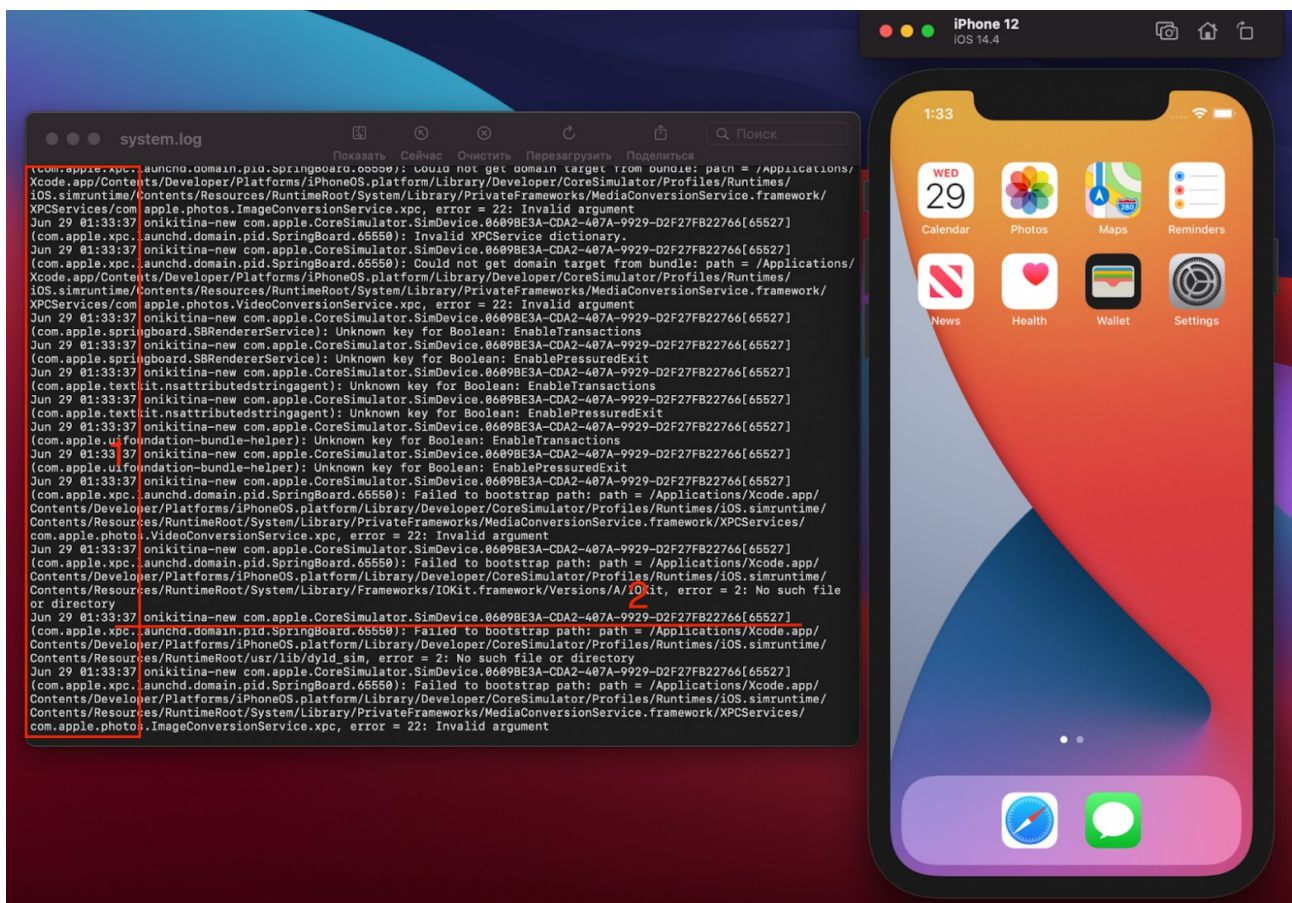
## iOS

В первую очередь нас интересует xCode — интегрированная среда разработки (IDE), в которую встроен нужный нам инструмент Simulator.

Как использовать инструмент:

1. Устанавливаем xCode.
2. В системной строке нажимаем xCode → Open Developer Tools → Simulator.
3. Устанавливаем приложение.
4. В самом симуляторе выбираем Debug → Open System Log.

Мы будем видеть логи в реальном времени:



Подобное оформление логов мы уже где-то видели, но построение информации в выдаче немного отличается. Есть дата и время (1) и данные (2) о том, с какого устройства снята информация: имя компьютера, элемент системы, с которого пришло сообщение, и его расшифровка.

Но первый способ работает только с симуляторами. Если необходимо снимать логи с реального устройства, в этом может помочь раздел Devices and Simulators.

Записи можно отфильтровать по конкретному процессу (вашему приложению):

1. Устанавливаем xCode.
2. Подключаем устройство к ПК по USB.
3. Открываем xCode → Windows → Devices and Simulators.

Дальше нажимаем у устройства Open Console и видим панель с названием устройства, информацией о модели и ОС:

I-P-040

iOS 14.3 (18C66)

Model: iPhone 12 Pro Max

Capacity: 112,45 GB (98,82 GB available)

Serial Number: [REDACTED]

Identifier: [REDACTED]

☒ Show as run destination

☐ Connect via network

Take Screenshot

View Device Logs

Open Console

PAIRED WATCHES

Name	Model	watchOS	Identifier

INSTALLED APPS

Name	1	Version	2	Identifier	3
[REDACTED]		113		[REDACTED]	
[REDACTED]		.. 1		[REDACTED]	
[REDACTED]		4		[REDACTED]	
[REDACTED]		40		[REDACTED]	
[REDACTED]		1		[REDACTED]	
[REDACTED]		125		[REDACTED]	
[REDACTED]		58		[REDACTED]	

+

-

⚙

1 — все приложения, которые установлены на устройстве, 2 — версия устройства, 3 — пакет приложения устройства.

Логи поступают в реальном времени, но их удобно отслеживать:

Консоль

10 357 сообщения

Приостановить

Сейчас

Активность

Очистить

Перезагрузить

Информация

Поделиться

🔍 Поиск

☐ Все сообщения

Ошибки и сбои

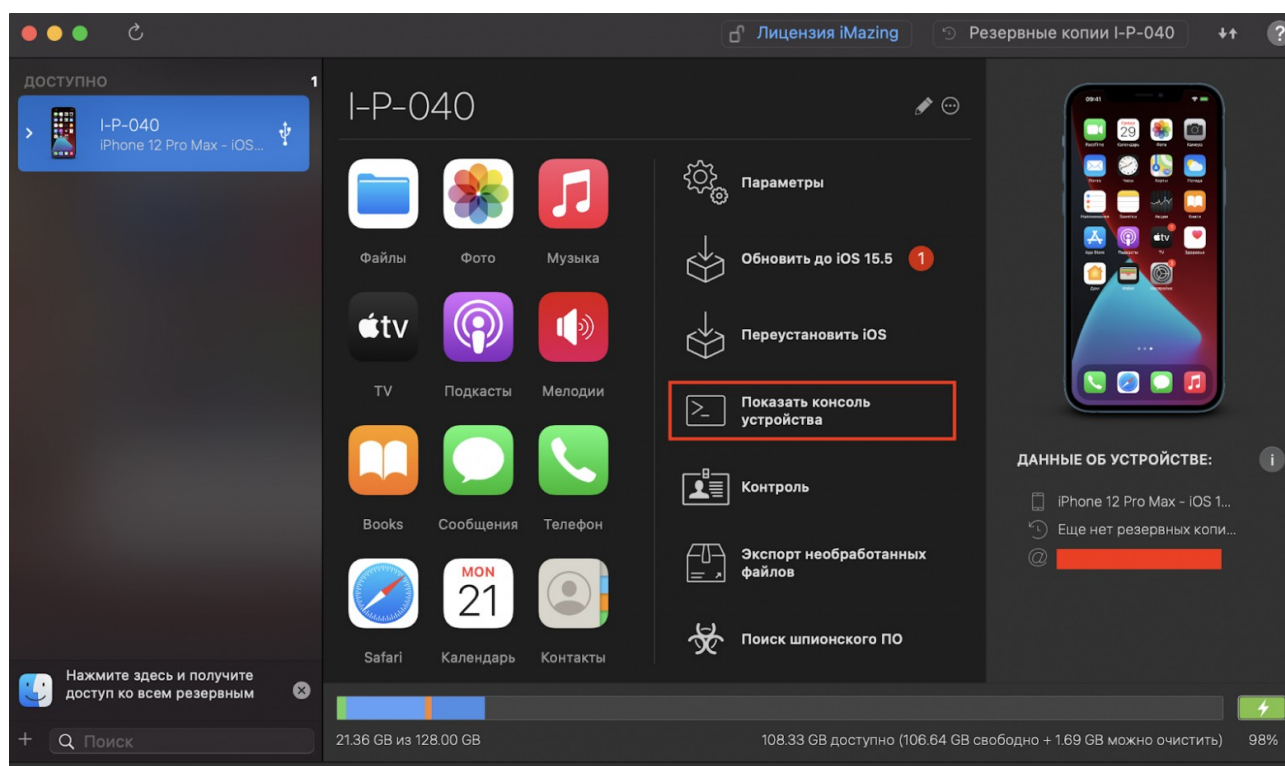
Тип	Время	Процесс	Сообщение
mDNSResp	01:40:31.801143+0300	[R0->Q65535] mDNSCoreReceiveNoUnicastAnswers: Removing expired record<mask.hash: 'I1tw3UPD/4+D6JJX/ndYg==>	
mDNSResp	01:40:31.801305+0300	[R18063->Q37252] getaddrinfo result -- event: add, ifindex: 0, name: <mask.hash: 'ZKd654Bw5bXaKPvEOCHW6g==>, type: A, rdati	
mDNSResp	01:40:31.801387+0300	[R18063->Q37252] getaddrinfo result -- event: add, ifindex: 0, name: <mask.hash: 'ZKd654Bw5bXaKPvEOCHW6g==>, type: A, rdati	
mDNSResp	01:40:31.801516+0300	[R18063->Q37252] getaddrinfo result -- event: add, ifindex: 0, name: <mask.hash: 'ZKd654Bw5bXaKPvEOCHW6g==>, type: A, rdati	
mDNSResp	01:40:31.801600+0300	[R18063->Q37252] getaddrinfo result -- event: add, ifindex: 0, name: <mask.hash: 'ZKd654Bw5bXaKPvEOCHW6g==>, type: A, rdati	
mDNSResp	01:40:31.801681+0300	[R18063->Q37252] getaddrinfo result -- event: add, ifindex: 0, name: <mask.hash: 'ZKd654Bw5bXaKPvEOCHW6g==>, type: A, rdati	
mDNSResp	01:40:31.801759+0300	[R18063->Q37252] getaddrinfo result -- event: add, ifindex: 0, name: <mask.hash: 'ZKd654Bw5bXaKPvEOCHW6g==>, type: A, rdati	
mDNSResp	01:40:31.801836+0300	[R18063->Q37252] getaddrinfo result -- event: add, ifindex: 0, name: <mask.hash: 'ZKd654Bw5bXaKPvEOCHW6g==>, type: A, rdati	
mDNSResp	01:40:31.801913+0300	[R18063->Q37252] getaddrinfo result -- event: add, ifindex: 0, name: <mask.hash: 'ZKd654Bw5bXaKPvEOCHW6g==>, type: A, rdati	
symptoms	01:40:31.806061+0300	Data Usage for com.apple.reminders - WiFi in/out: 195060/120993, WiFi delta_in/delta_out: 198/70, Cell in/out: 0/0, Cell de	
mDNSResp	01:40:31.806241+0300	[R18063] getaddrinfo stop -- hostname: <mask.hash: 'HmR8lUF67bFLwscJfMO7A==>, client pid: 138 (cloudd)	
cloudd	01:40:31.807084+0300	[C2841 Hostname#add73bf7:443 in_progress resolver (satisfied (Path is satisfied), interface: en0, ipv4, dns)] event: resolv	
cloudd	01:40:31.815917+0300	tcp_input [C2841.1:3] flags=[S.E] seq=2132255458, ack=120131855, win=65160 state=SYN_SENT rcv_nxt=0, snd_una=120131854	
cloudd	01:40:31.816030+0300	nw_flow_connected [C2841.1 IPv4#bf760ad1:443 in_progress channel-flow (satisfied (Path is satisfied), viable, interface: en0	
cloudd	01:40:31.816318+0300	[C2841.1 IPv4#bf760ad1:443 in_progress channel-flow (satisfied (Path is satisfied), viable, interface: en0, ipv4, dns)] eve	
cloudd	01:40:31.816437+0300	[C2841 Hostname#add73bf7:443 in_progress resolver (satisfied (Path is satisfied), interface: en0, ipv4, dns)] event: flow:f	
cloudd	01:40:31.818155+0300	boringsl_session_apply_protocol_options_for_transport_block_invoke(1689) [C2841.1:2][0x1190efd70] TLS configured [min_vers:	
cloudd	01:40:31.818535+0300	boringsl_context_info_handler(1821) [C2841.1:2][0x1190efd70] Client handshake started	
cloudd	01:40:31.819067+0300	boringsl_context_info_handler(1836) [C2841.1:2][0x1190efd70] Client handshake state: TLS client enter_early_data	
cloudd	01:40:31.819490+0300	boringsl_context_info_handler(1836) [C2841.1:2][0x1190efd70] Client handshake state: TLS client read_server_hello	

У нас есть три столбца:

- 1.«Время» — время поступления сообщения.
- 2.«Процесс» — с какой части системы/приложения пришло сообщение.
- 3.«Сообщение» — описание события, сервисная информация.

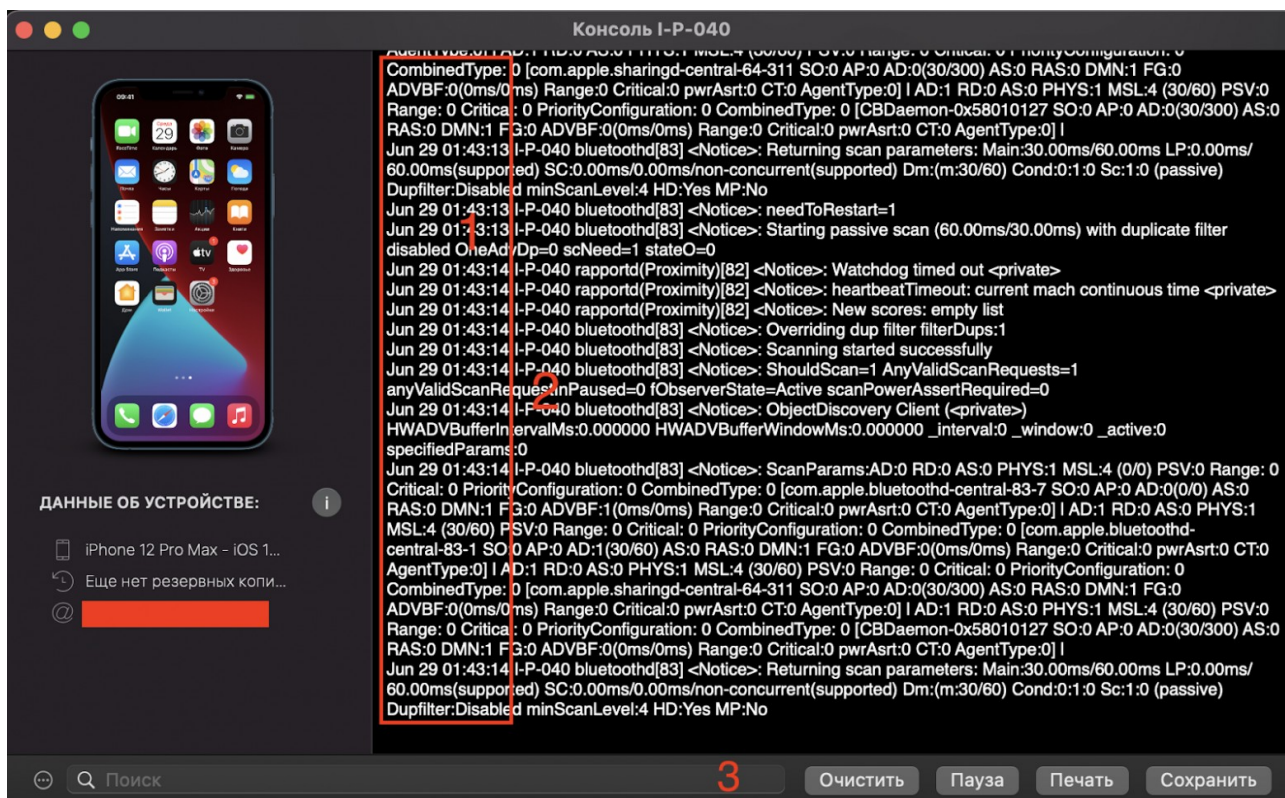
В инструменте есть поиск для фильтрации выдачи. Ещё есть полезная кнопка «Приостановить» — она останавливает поток логов.

А вот утилита iMazing поможет снимать iOS-логи для тех, у кого установлен Windows. Приложение платное, но часть функциональности доступна бесплатно. Например, за снятие логов устройства платить не нужно.



В меню выбираем «Показать консоль устройства». В открывшемся окне приходят записи логов в реальном времени со всего устройства.





1 — дата и время получения сообщения; 2 — имя телефона, информация, с какой части устройства пришло сообщение, и описание; 3 — поисковая строка для фильтрации выдачи.

Ещё одно важное достоинство iMazing — возможность сохранять логи (разумеется, по кнопке «Сохранить»).