



HTTP API ПК Интеллект

1. Общие сведения о HTTP API	3
2. Версия продукта	3
3. Карта	4
3.1 Получение списка карт	4
3.2 Информация об одной карте	4
3.3 Список слоёв для выбранной карты	5
3.4 Информация о конкретном слое	6
3.5 Фоновый рисунок слоя	6
3.6 Список точек на слое	6
3.7 Информация об отдельной точке на слое	9
4. Классы объектов	9
4.1 Список классов объектов, которые существуют на сервере	9
4.2 Отдельный класс объектов	10
4.3 Список состояний для определённого класса объектов	10
4.4 Информация о конкретном состоянии	10
4.5 Получение иконки для определённого состояния	11
5. Объекты	11
5.1 Получение списка объектов	11
5.2 Информация об отдельном объекте	12
5.3 Состояние отдельного объекта	13
5.4 Список доступных действий с объектом, находящимся в определённом состоянии	13
6. Получение событий	14
6.1 Получение событий видеоподсистемы блоками	15
7. Отсылка команд на сервер	17
8. Макрокоманды	17
9. Видео	18
9.1 Запрос миниатюр (скриншотов)	18
9.2 Запрос конфигурации	19
9.3 Запрос видео	21
9.4 Формат основного потока	21
9.4.1 Управление записью	23
9.4.2 Постановка и снятие с охраны камеры	23
9.4.3 Управление телеметрией	24
9.4.4 Работа с архивом	24
10. Нотификация	29
11. Звук	31
11.1 Получение живого звука	31
11.2 Проигрывание звука из архива	32
11.3 Отправка живого звука	33

Общие сведения о HTTP API

Программно HTTP API предоставляется модулем web2 (*Веб-сервер 2.0*).



Примечание.

См. Руководство Администратора, раздел Настройка Сервера для подключения Клиентов с помощью модуля Веб-сервер 2.0.

Port – порт.

/somecontext – опциональный веб-контекст, в котором работает приложение. Это контекст веб-приложения.

Таким образом можно на одном домене иметь несколько систем:

www.example.com/sistema1/

www.example.com/videosistema23/

www.example.com/a/

Причем этот контекст может быть более сложным:

www.example.com/redirects/toitvwebserver/firstsystem/

www.example.com/redirects/toitvwebserver/secondsystem/

www.example.com/redirects/toitvwebserver/sauna/

Далее описание будет опускаться там, где действие запроса понятно из контекста.



Внимание!

URL, id объектов и расширения файлов чувствительны к регистру.



Примечание.

Дата и время везде используется в формате RFC3339, подробнее см. <http://www.ietf.org/rfc/rfc3339.txt>

Версия продукта

Для идентификации сервера можно использовать URL

`http://example.com:[port][/somecontext]/product/version`

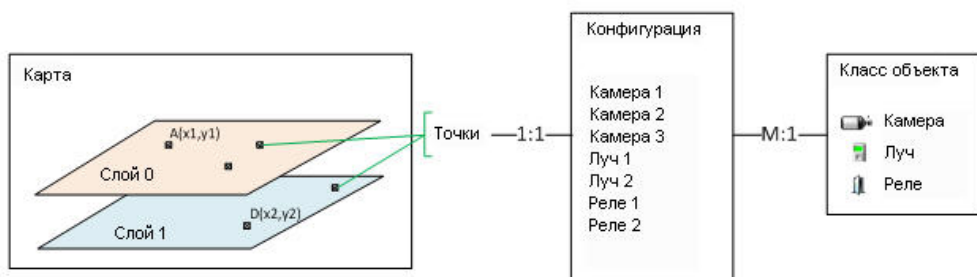
Если в ответ приходит text/plain строка типа

Intellect 4.8.4

Это означает, что сервер поддерживает протокол, описанный в данном документе. Строка может меняться в зависимости от версии продукта. Это сделано для того, чтобы

различать два схожих по функциональным возможностям, но разных по протоколу веб-сервера в разных продуктах.

Карта



На Сервере может быть создано несколько карт. Каждая карта может содержать один и более слоёв. На каждом слое расположены точки. Каждая точка связана с одним из объектов конфигурации.

Конфигурация – это объекты ПК *Интеллект*. Каждый объект является объектом определённого класса. Каждый объект имеет одно состояние и список действий, которые можно с ним производить.

Класс объекта описывает его вид (значки), возможные состояния и возможные действия с объектом в каждом из состояний .

Получение списка карт

Карт может быть 0 и более.

[http://example.com:\[port\]\[somecontext\]/secure/kartas/](http://example.com:[port][somecontext]/secure/kartas/)

Пример ответа:

```
<kartas>
<karta>
<id>plan</id>
<name>This is plan of a building</name>
</karta>
<karta>
<id>site</id>
<name>This is site around the building</name>
</karta>
</kartas>
```

Информация об одной карте

plan – id карты.

http://example.com:[port][/somecontext]/secure/kartas/**plan**/

Пример ответа:

```
<karta>
<id>plan</id>
<name>This is plan of a building</name>
</karta>
```

Список слоёв для выбранной карты

plan – id карты.

Слоёв может быть 1 и более.

http://example.com:[port][/somecontext]/secure/kartas/**plan**/layers/

Пример ответа:

```
<layers>
<layer>
<height>1000</height>
<id>base</id>
<mapId>plan</mapId>
<name>Base layer for plan</name>
<width>1000</width>
<zoomDef>1.0</zoomDef>
<zoomMax>4.0</zoomMax>
<zoomMin>0.25</zoomMin>
<zoomStep>0.25</zoomStep>
</layer>
</layers>
```

Описание параметров:

Height – высота картинки слоя в пикселях;

Width – ширина картинки слоя в пикселях;

zoomMin – минимальный масштаб картинки;

zoomMax – максимальный масштаб картинки;

zoomStep – шаг увеличения масштаба при zoom in и zoom out;

zoomDef – масштаб по-умолчанию.

Пример. Пусть ширина картинки равна 100 пикселям. Тогда ширина для масштаба 0,25 будет $100 * 0,25 = 25$ пикселей.

Информация о конкретном слое

Описание параметров см. в разделе [Список слоёв для выбранной карты](#).

base – id слоя.

[http://example.com:\[port\]\[somecontext\]/secure/kartas/plan/layers/base/](http://example.com:[port][somecontext]/secure/kartas/plan/layers/base/)

Пример ответа:

```
<layer>
  <height>1000</height>
  <id>base</id>
  <mapId>plan</mapId>
  <name>Base layer for plan</name>
  <width>1000</width>
  <zoomDef>1.0</zoomDef>
  <zoomMax>4.0</zoomMax>
  <zoomMin>0.25</zoomMin>
  <zoomStep>0.25</zoomStep>
</layer>
```

Фоновый рисунок слоя

[http://localhost:8080/server-1.0/secure/kartas/plan/layers/base/image.\[png|jpg\]](http://localhost:8080/server-1.0/secure/kartas/plan/layers/base/image.[png|jpg])

В ответ приходит изображение в формате png или jpg.

На запрос JPG, Jpg, JPEG, PNG будет возвращаться ошибка 404

Список точек на слое

[http://example.com:\[port\]/\[somecontext\]/secure/kartas/plan/layers/base/points/](http://example.com:[port]/[somecontext]/secure/kartas/plan/layers/base/points/)

id – совпадает с id объекта из конфигурации. Id не обязательно всегда равен CAM:1. Следует воспринимать id как строку.

Координатная сетка привязана к слою следующим образом:



Т.е. x и y не могут быть отрицательными, но могут быть дробными.

Пример ответа:

```
<points>
  <point>
    <id>CAM:1</id>
    <layerId>base</layerId>
    <mapId>plan</mapId>
    <x>100.0</x>
    <y>100.0</y>
  </point>
  <point>
    <id>CAM:2</id>
    <layerId>base</layerId>
    <mapId>plan</mapId>
    <x>200.0</x>
```

```
<y>200.0</y>
</point>
<point>
  <id>GRAY:1</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>300.0</x>
  <y>300.0</y>
</point>
<point>
  <id>GRAY:2</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>400.0</x>
  <y>400.0</y>
</point>
<point>
  <id>GRELE:1</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>500.0</x>
  <y>500.0</y>
</point>
<point>
  <id>GRELE:2</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>600.0</x>
  <y>600.0</y>
```



```
</point>
</points>
```

Информация об отдельной точке на слое

[http://example.com:\[port\]/somecontext/secure/kartas/plan/layers/base/points/CAM:2](http://example.com:[port]/somecontext/secure/kartas/plan/layers/base/points/CAM:2) – запрос информации о точке, соответствующей камере с идентификатором 2.

Пример ответа:

```
<point>
  <id>CAM:2</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>200.0</x>
  <y>200.0</y>
</point>
```

Классы объектов

Список классов объектов, которые существуют на сервере

[http://example.com:\[port\]/somecontext/secure/objectClasses](http://example.com:[port]/somecontext/secure/objectClasses)

Пример ответа:

```
<objectClasses>
  <objectClass>
    <id>GRELE</id>
  </objectClass>
  <objectClass>
    <id>USERS</id>
  </objectClass>
  <objectClass>
    <id>CAM</id>
  </objectClass>
  <objectClass>
```

```
    <id>RIGHTS</id>
  </objectClass>
  <objectClass>
    <id>GRAY</id>
  </objectClass>
</objectClasses>
```

Отдельный класс объектов

[http://example.com:\[port\]/\[somecontext\]/secure/objectClasses/GRELE/](http://example.com:[port]/[somecontext]/secure/objectClasses/GRELE/)

Пример ответа:

```
<objectClass>
  <id>GRELE</id>
</objectClass>
```

Список состояний для определённого класса объектов

[http://example.com:\[port\]/\[somecontext\]/secure/objectClasses/GRELE/states/](http://example.com:[port]/[somecontext]/secure/objectClasses/GRELE/states/) - получить список состояний для класса объектов **Реле**.

Пример ответа:

```
<states>
  <state>
    <id>off</id>
  </state>
  <state>
    <id>on</id>
  </state>
  <state>
    <id>disabled</id>
  </state>
</states>
```

Информация о конкретном состоянии

[http://example.com:\[port\]/somecontext/secure/objectClasses/\[ObjectClass\]/states/\[State\]/](http://example.com:[port]/somecontext/secure/objectClasses/[ObjectClass]/states/[State]/)

Пример:

[http://example.com:\[port\]/somecontext/secure/objectClasses/GRELE/states/off/](http://example.com:[port]/somecontext/secure/objectClasses/GRELE/states/off/) - получение информации о состоянии OFF класса объектов **Реле**.

Пример ответа:

```
<state>
<id>off</id>
</state>
```

Получение иконки для определённого состояния

[http://example.com:\[port\]/somecontext/secure/objectClasses/\[ObjectClass\]/states/\[State\]/image.png](http://example.com:[port]/somecontext/secure/objectClasses/[ObjectClass]/states/[State]/image.png)

Пример:

[http://example.com:\[port\]/somecontext/secure/objectClasses/GRELE/states/off/image.png](http://example.com:[port]/somecontext/secure/objectClasses/GRELE/states/off/image.png) - получение иконки для состояния OFF класса объектов **Реле**.

В ответ приходит изображение в формате png:



Объекты

Получение списка объектов

[http://example.com:\[port\]/somecontext/secure/configuration/](http://example.com:[port]/somecontext/secure/configuration/)

JSON

```
[ {
  "type" : "CAM",
  "id" : "CAM:2",
  "extId" : "2",
  "name" : "Camera 2",
  "regionId" : "2.1",
  "state" : {
    "id" : "alarmed",
    "type" : "ALARM"
  },
  "presets" : [ ]
}, {
  "type" : "CAM",
  "id" : "CAM:1",
  "extId" : "1",
```

```

    "name" : "Camera 1",
    "state" : {
      "id" : "armed",
      "type" : "NORMAL"
    },
    "presets" : [ ]
  }, {
    "type" : "GRAY",
    "id" : "GRAY:1",
    "extId" : "1",
    "name" : "Sensor 1",
    "state" : {
      "id" : "disconnected",
      "type" : "ALARM"
    }
  }, {
    "type" : "GRELE",
    "id" : "GRELE:2",
    "extId" : "2",
    "name" : "Relay 2",
    "state" : {
      "id" : "disabled",
      "type" : "NORMAL"
    }
  }, {
    "type" : "GRELE",
    "id" : "GRELE:1",
    "extId" : "1",
    "name" : "Relay 1",
    "regionId" : "2.1",
    "state" : {
      "id" : "disabled",
      "type" : "NORMAL"
    }
  }, {
    "type" : "GRAY",
    "id" : "GRAY:2",
    "extId" : "2",
    "name" : "Sensor 2",
    "state" : {
      "id" : "disconnected",
      "type" : "ALARM"
    }
  }
]

```

Информация об отдельном объекте

[http://example.com:\[port\]/somecontext/secure/configuration/GRAY:2/](http://example.com:[port]/somecontext/secure/configuration/GRAY:2/) – получение информации об объекте **Луч** с идентификатором 2.

Пример ответа:

```
<GRAY>
  <id>GRAY:2</id>
  <name> 2</name>
  <state>
    <id>alarmed</id>
  </state>
</GRAY>
```

Состояние отдельного объекта

[http://example.com:\[port\]/\[somecontext\]/secure/configuration/GRAY:2/state/](http://example.com:[port]/[somecontext]/secure/configuration/GRAY:2/state/) – получение состояния объекта **Луч** с идентификатором 2.

Пример ответа:

```
<state>
<id>alarmed</id>
</state>
```

Список доступных действий с объектом, находящимся в определённом состоянии

Список действий запрашивается не по классу объекта, а берётся из контекста конкретного объекта, т.к. возможны различные права пользователя на объекты одного и того же класса.

Работа с полученным списком описана в разделе [Отсылка команд на сервер](#).

[http://example.com:\[port\]/\[somecontext\]/secure/configuration/GRAY:2/state/actions/](http://example.com:[port]/[somecontext]/secure/configuration/GRAY:2/state/actions/) – получение списка доступных действия для объекта **Луч** с идентификатором 2.

Пример ответа:

```
<actions>
  <action>
    <description>Disarm ray</description>
    <id>ray.disarm</id>
  </action>
  <action>
    <description>Confirm alarm</description>
    <id>ray.confirm</id>
```

```
</action>
</actions>
```

Если состояние объекта не предусматривает никаких действий, то xml будет таким:

```
<actions/>
```

Получение событий

Соединение не разрывается и события приходят бесконечно.

action – тип события. Возможные значения: create, delete, update.

Все поля ниже опциональны:

objectId – id объекта, от которого приходит событие (обязательно приходит с update, delete, create).

state – id нового состояния объекта (обязательно приходит в create. Если состояние не изменилось, то в событии update состояния не будет).

x, y – новые координаты, если изменились.

Запрос:

```
http://example.com:[port][/somecontext]/secure/feed/
```

Примеры ответа:

```
<message>
  <action>update</action>
  <objectId>CAM:1</objectId>
  <state>disconnected</state>
</message>
```

```
<message>
  <action>state</action>
  <objectId>CAM:1</objectId>
  <x>10.0</x>
  <y>123.9</y>
</message>
```

```
<message>
  <action>state</action>
  <objectId>CAM:1</objectId>
  <state>connected</state>
  <x>300.8</x>
  <y>670</y>
</message>
```

```
<message>
  <action>state</action>
  <objectId>CAM:1</objectId>
  <x>100</x>
  <y>100</y>
</message>
```

```
<message>
  <action>ping</action>
</message>
```

Получение событий видеоподсистемы блоками

[http://example.com:\[port\]\[somecontext\]/secure/events/](http://example.com:[port][somecontext]/secure/events/)

Параметры:

from – Самая старая дата промежутка поиска сообщений. (2012-12-27T15%3A19%3A16.000%2B04%3A00)

to – Самая последняя дата промежутка поиска сообщений. (2012-12-27T15%3A19%3A16.000%2B04%3A00)

count – максимальное количество сообщений в ответе [1, 200]. По-умолчанию 20. Сервер может вернуть чуть больше, если сообщений в базе данных осталось мало.

objectId – id объекта (CAM:1, GRAY:5 и т.д). Если параметр не задан, то возвращаются события всех объектов.

Коды возврата:

200 - ОК

400 - неверный параметр (формат даты, например)

500 - ошибка

503 - ошибка соединения с ядром

504 - таймаут (ядро не вернуло данные в течение 2000 миллисекунд)

Примеры ответа

XML:

```
<events>
  <event>
    <description> </description>
    <id>{E56B09A0-1A50-E211-840E-005056C00008}</id>
    <objectId>CAM:1</objectId>
    <ts>2012-12-27T15:43:27+04:00</ts>
  </event>
  <event>
    <description> </description>
    <id>{4482F63F-1A50-E211-840E-005056C00008}</id>
    <objectId>CAM:1</objectId>
    <ts>2012-12-27T15:40:50+04:00</ts>
  </event>
  <event>
    <description> </description>
    <id>{35D4BE3E-1750-E211-840E-005056C00008}</id>
    <objectId>CAM:1</objectId>
    <ts>2012-12-27T15:19:16+04:00</ts>
  </event>
</events>
```

JSON:

```
[ {
  "id" : "{E56B09A0-1A50-E211-840E-005056C00008}",
  "description" : " ",
  "ts" : "2012-12-27T15:43:27.000+04:00",
```



```
"objectId" : "CAM:1"
}, {
  "id" : "{4482F63F-1A50-E211-840E-005056C00008}",
  "description" : " ",
  "ts" : "2012-12-27T15:40:50.000+04:00",
  "objectId" : "CAM:1"
}, {
  "id" : "{35D4BE3E-1750-E211-840E-005056C00008}",
  "description" : " ",
  "ts" : "2012-12-27T15:19:16.000+04:00",
  "objectId" : "CAM:1"
} ]
```

Отсылка команд на сервер

PUT

[http://example.com:\[port\]/\[somecontext\]/secure/configuration/GRAY:2/state/actions/disarm/execute](http://example.com:[port]/[somecontext]/secure/configuration/GRAY:2/state/actions/disarm/execute) - пример отсылки на сервер команды снятия с охраны Луча с идентификатором 2.

Макрокоманды

В разделе:

- [Получение списка макрокоманд](#)
- [Получение параметров макрокоманд](#)
- [Запрос на выполнение макрокоманды на сервере](#)

Макрокоманды (макросы) – это некоторая предопределённая последовательность реакций на определённые события. Макрокоманды создаются на сервере и имеют ID и название. Они похожи на действия с объектами, но не привязаны к объекту.

Получение списка макрокоманд

GET

[http://example.com:\[port\]/\[somecontext\]/secure/actions/](http://example.com:[port]/[somecontext]/secure/actions/)

Пример ответа:

```
<actions>
  <action>
    <description>Start recording by all cameras</description>
    <id>macro2</id>
  </action>
  <action>
    <description>Disarm all zones</description>
    <id>1</id>
  </action>
</actions>
```

Получение параметров макрокоманд

Каких-либо дополнительных параметров у объекта нет. Можно ограничиться получением списка макросов.

GET

[http://example.com:\[port\]/\[somecontext\]/secure/actions/macro2/](http://example.com:[port]/[somecontext]/secure/actions/macro2/) - получение параметров макрокоманды с идентификатором macro2.

Пример ответа:

```
<action>
<description>Start recording by all cameras</description>
<id>macro2</id>
</action>
```

Запрос на выполнение макрокоманды на сервере

PUT

[http://example.com:\[port\]/\[somecontext\]/secure/actions/macro2/execute](http://example.com:[port]/[somecontext]/secure/actions/macro2/execute) – запрос на выполнение на сервере макрокоманды с идентификатором macro2.

Видео

Запрос миниатюр (скриншотов)

[http://example.com:\[port\]/\[somecontext\]/secure/video/image.jpg?cam.id=1&version=4.7.8.0](http://example.com:[port]/[somecontext]/secure/video/image.jpg?cam.id=1&version=4.7.8.0)

Параметры:

cam.id – обязательный. Id камеры.

width -- необязательный. Значение может быть в диапазоне [64, 1600]. Сервер автоматически округляет ширину до большего значения, кратного 4.

height – необязательный. Значение может быть в диапазоне [30, 1200].

version – необязательный. версия клиента (на случай смены протокола). Сейчас нужно посылать значение "4.7.8.0".

login – необязательный. Логин;

password – необязательный. Если установлен доступ по паролю.

если width и height не установлены то размер возвращаемого изображения берётся из видеопотока.

Возвращаемое значение:

изображение jpeg приблизительно запрошенного размера.

Если произошла ошибка, то возвращается http код ошибки:

404 – камера отключена или не используется (disabled);

403 – неверный пароль;

426 – старая версия клиента;

429 – слишком много запросов;

444 – потерян сигнал по камере или камера отключена (коаксиальный провод отключен от платы);

503 – ошибка архива.

Пример:

Получить изображение с камеры 5, шириной приблизительно 85 пикселей:

```
http://localhost:8079/web2/secure/video/image.jpg?cam.id=5&width=85&version=4.7.8.0&login=username&password=secret
```

В ответ придёт картинка jpg шириной 88 пикселей, либо код об ошибке и body нулевой длины (т.е. придут только заголовки).

Запрос конфигурации

GET

```
http://example.com:[port]/[somecontext]/secure/video/config.properties?version=4.7.8.0&login=XXX&password=YYY
```

Параметры:

- version – обязательное поле. Версия клиента (на случай смены протокола). Сейчас нужно посылать значение "4.7.8.0".
- login – необязательное поле. Логин.
- password – необязательное поле. Используется, если установлен доступ по паролю.

Особенности использования

В начале работы неизвестно, установлены ли пароль, логин и т.п. Поэтому в первый раз необходимо послать следующий запрос:

GET

```
http://www.examplehost.com/[somecontext]/secure/video/config.properties?version=4.7.8.0
```

В ответ сервер отправит текстовый файл config.properties следующего формата:

```
password.enabled=true
```

```
login.enabled=true
```

```
password.invalid=true#
```



**Примечание.**

Символ # является признаком конца конфигурационного файла.

После получения файла такого вида можно понять, что пароль установлен и пароль неправильный. Неправильный он потому, что в данном случае был послан пустой пароль и пустой логин.

Необходимо запросить у пользователя логин и пароль и снова отослать серверу запрос на конфигурацию:

GET

[http://www.examplehost.com\[/somecontext\]/secure/video/config.properties?version=4.7.8.0&login=XXX&password=YYY](http://www.examplehost.com[/somecontext]/secure/video/config.properties?version=4.7.8.0&login=XXX&password=YYY)

Если пароль правильный или доступ разрешен без пароля, то сервер в ответ вышлет конфигурацию в следующем виде:

```
password.enabled=true
login.enabled=true
password.invalid=false
cam.0.id=2
cam.0.name=Face
cam.0.rights=11
cam.1.id=3
cam.1.name=Camera 3
cam.1.rights=11
cam.2.id=5
cam.2.name=Camera 5
cam.2.rights=11
cam.2.telemetry_id=1.1
cam.count=3#
```

password.invalid=false означает, что введён верный пароль.

**Примечание.**

Если разрешен доступ без пароля, то password.enabled=false, и вся нужная конфигурация будет получена с первого раза.

cam.count=3 – общее количество камер в присланной конфигурации (id начинается с нуля).

Для каждой из трёх камер необходимо получить данные из конфигурации.

cam.N.id – id камеры.

cam.N.name – название камеры.

cam.N.rights – права.

cam.N.telemetry_id – id телеметрии (может отсутствовать, если телеметрии нет, тогда необходимо скрывать элементы управления телеметрией).

cam.N.rights – определяет права (они проверяются на сервере, но чтобы не показывать пользователю лишних опций, доступны и на клиенте). Параметр представляет собой флаги. Если флаг проставлен, то элемент интерфейса следует показывать, если нет, то скрывать.

```
static final int RIGHT_VIEW = 0x1; // доступен просмотр живого видео (этот всегда проставлен в 1)
```

```
static final int RIGHT_CONTROL = 0x2; // управление (телеметрия, постановка и снятие с охраны)
```

```
static final int RIGHT_CONFIG = 0x4; // reserved
```

```
static final int RIGHT_HISTORY = 0x8; // доступ к архиву
```

Запрос видео

[http://example.com:\[port\]\[somecontext\]/secure/video/action.do?version=4.7.8.0&sessionId=FC126734&cam.id=5&login=XXX&password=YYY](http://example.com:[port][somecontext]/secure/video/action.do?version=4.7.8.0&sessionId=FC126734&cam.id=5&login=XXX&password=YYY) - запрос видео для камеры с идентификатором 5.

cam.id – идентификатор камеры.

sessionId – любое значение.

Формат основного потока

В ответе приходит поток в виде:

```
HTTP/1.0 200 OK
Connection: close
Server: ITV-Intellect-Webserver/4.9.0.0
Cache-Control: no-store,no-cache,must-revalidate,max-age=0
Pragma: no-cache
Date: Mon, 13 Jan 2013 10:44:27 GMT
Content-Type: multipart/mixed;boundary=videoframe

--videoframe
Content-Type: text/xml
Content-Length: 138

<video_in>
  <sessionid>FC126734</sessionid>
  <video_in>CAM:5</video_in>
  <newstate>started</newstate>
  <errcode>100</errcode>
</video_in>
--videoframe
Content-Type: image/jpeg
Content-Length: 23978
X-Width: 320
X-Height: 240
X-Time: 2013-03-15T10:51:44.314+04:00
X-Timestamp: 0.000

  <jpeg image>
--videoframe
Content-Type: image/jpeg
Content-Length: 23651
X-Width: 320
X-Height: 240
X-Time: 2013-03-15T10:51:44.314+04:00
X-Timestamp: 0.152

  <jpeg image>
```

Здесь:

- X-Width - ширина изображения.
- X-Height- высота изображения.
- X-Time - абсолютное время формирования фрейма.
- X-Timestamp - относительное время фрейма в секундах (относительно начала потока).

В случае завершения потока по вине сервера может прийти завершающий пакет:

```
--videoframe
Content-Type: text/xml
Content-Length: 106
<video_in>
  <sessionid>FC126734</sessionid>
  <video_in>CAM:5</video_in>
  <newstate>closed</newstate>
  <errcode>103</errcode>
</video_in>
```

- sessionid - id сессии (тот же что и при старте).
- video_in - идентификатор камеры.
- errcode - код ошибки:
 - 100 – отсутствие ошибки.
 - 101 – слишком много подключенных пользователей.
 - 102 – неверный пароль (пароль, теоретически, могут поменять в любой момент работы).
 - 103 – видео недоступно.
 - 104 – старая версия клиента. Обновите версию.

Управление записью

Начало записи

GET

`http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=1&target=CAM&targetid=1&command=REC&login=XXX&password=YYY`

Окончание записи

GET

`http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&target=CAM&targetid=1&command=REC_STOP&login=XXX&password=YYY`

Здесь targetid==cam.id.

Постановка и снятие с охраны камеры

Постановка на охрану

GET

`http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=1&target=CAM&targetid=1&command=ARM&login=XXX&password=YYY`

Снятие с охраны

GET

`http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&target=CAM&targetid=1&command=DISARM&login=XXX&password=YYY`

Здесь targetid==cam.id.

Управление телеметрией

GET

[http://example.com:\[port\]/\[somecontext\]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&target=PTZ&targetid=1.1&command=RIGHT&login=XXX&password=YYY&speed=2](http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&target=PTZ&targetid=1.1&command=RIGHT&login=XXX&password=YYY&speed=2)

Параметры:

command – обязательный параметр. Может принимать следующие значения:

- RIGHT
- UP
- LEFT
- DOWN
- ZOOM_IN
- ZOOM_OUT
- GO_PRESET – перейти в определенный пресет.
- POINTMOVE – зуммирование выделенной точки на изображении (x, y).
- AREAZOOM – зуммирование выделенной области изображения (x,y,w,h).

speed – обязательный параметр. Скорость отработки команды (от 0 до 10). При управлении по сети из-за задержек лучше использовать низкие значения.

cam.id – обязательный параметр. Идентификатор камеры.

target – обязательный параметр. Всегда равно PTZ.

targetid – обязательный параметр. Id телеметрии, связанной с камерой (присылается в конфигурации SETUP).

preset – номер пресета (число). Обязательный параметр только для command=GO_PRESET. В остальных случаях его значение игнорируется.

x – координата x относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=POINTMOVE или command=AREAZOOM. В остальных случаях его значение игнорируется.

y – координата y относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=POINTMOVE или command=AREAZOOM. В остальных случаях его значение игнорируется.

w – ширина области зуммирования y относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=AREAZOOM. В остальных случаях его значение игнорируется.

h – высота области зуммирования относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=AREAZOOM. В остальных случаях его значение игнорируется.

Работа с архивом

В разделе:

- Получение списка записей - "arc.intervals"
- Получение одного фрейма из архива - "arc.frame"
- Получение видео из архива - "arc.play"
- Получение списка записей (2-й способ)

Поток из видеоархива присылается в таком же формате, что и живое видео.

Получение списка записей - "arc.intervals"

GET

`http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionId=29101F1&video_in=CAM:5&command=arc.intervals&time_from=2013-03-20T00:00:00.000+04:00&time_to=2013-03-22T23:59:59.999+04:00&max_count=100&split_threshold=10399&login=XXX&password=YYY`

Обязательные параметры:

`command=arc.intervals` – команда для получения списка записей

`video_in` – идентификатор камеры.

`time_from` – начало интересующего диапазона времени.

Необязательные параметры:

`time_to` – начало и конец интересующего диапазона времени.

`max_count` – максимальное количество записей в ответе

`split_threshold` – время для объединения нескольких интервалов (в секундах). Интервалы, расстояние между которыми будет меньше заданного, будут объединены в один.

В ответе придёт **XML**:

```
<?xml version="1.0" encoding="UTF-8"?>
<records>
  <record>
    <from>2011-09-01T00:00:00-05:00</from>
    <to>2011-09-01T00:00:35-05:00</to>
  </record>
  <record>
    <from>2011-09-01T00:00:35-05:00</from>
    <to>2011-09-01T00:01:10-05:00</to>
  </record>
</records>
```

Получение одного фрейма из архива - "arc.frame"

GET

`http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionId=29101F1&video_in=CAM:5&command=arc.frame&time=2013-03-22T13:04:52.312+04:`

00&range=0.1&login=XXX&password=YYY

Обязательные параметры:

command=arc.frame - команда для одного фрейма;

video_in - идентификатор камеры;

time - время, которое нас интересует.

Необязательные параметры:

range - время в секундах, для задания диапазона поиска ближайшего фрейма относительно time (если не указан, ищется ближайший по всему архиву);

imageWidth - ширина в пикселях (если не указано или 0, рассчитывается автоматически с сохранением пропорций);

imageHeight - высота в пикселях (если не указано или 0, рассчитывается автоматически с сохранением пропорций);

fps - максимальная частота кадров в секунду (если не указано или 0, часта кадров не будет ограничиваться).

В ответ придут http-заголовки и ближайший фрейм из диапазона [time - range, time + range] в формате jpeg. Если фрейма в диапазоне не будет тело в ответе будет пустым.

Получение видео из архива - "arc.play"

GET

http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionId=29101F1&video_in=CAM:5&command=arc.play&time_from=2013-03-22T13:04:52.312+04:00&time_to=2013-03-22T13:16:31.873+04:00&login=XXX&password=YYY

Обязательные параметры:

command=arc.play - команда для получения видео из архива;

video_in - идентификатор камеры;

time_from - время начала проигрывания архива.

Необязательные параметры:

time_to - время завершения проигрывания архива (если не указано, будет отдан весь архив до последней записи);

imageWidth - ширина в пикселях (если не указано или 0, рассчитывается автоматически с сохранением пропорций);

imageHeight - высота в пикселях (если не указано или 0, рассчитывается автоматически с сохранением пропорций);

fps - максимальная частота кадров в секунду (если не указано или 0, часта кадров не будет ограничиваться).

При завершении потока придет завершающий пакет с newstate=closed и errcode=100.

Получение списка записей (2-й способ)

GET

http://example.com:[port]/[somecontext]/secure/archive/**CAM:2/[2011-12-30|2011-12]?[splitThreshold=50]&[days=1]**

splitThreshold – если разница между окончанием предыдущей записи и началом следующей меньше этого числа (в миллисекундах), то записи объединяются в одну. Чтобы никакие записи не объединялись, нужно указать splitThreshold=0. [default: 50].

days - количество дней, от текущего, за которые требуется получить архив. [default: 1]

Всё время интерпретируется как локальное для сервера.

Получить записи за 18 ноября 2013 года и слепить все записи, промежуток между которыми меньше 2000 миллисекунд:

http://example.com:[port]/[somecontext]/secure/archive/**CAM:1/2013-10-18/?splitThreshold=2000**

Получить записи за 10 дней, начиная с 18 ноября 2013 года:

http://example.com:[port]/[somecontext]/secure/archive/**CAM:1/2013-10-18/?days=10**

XML:

```
<?xml version="1.0" encoding="UTF-16"?>
<days>
  <day>
    <id>2013-11-10T00:00:00-02:00</id>
    <records>
      <from>2013-11-10T18:44:01.579-02:00</from>
      <to>2013-11-10T18:44:09.717-02:00</to>
    </records>
  </day>
  <day>
    <id>2013-11-18T00:00:00-02:00</id>
    <records>
      <from>2013-11-18T18:38:30.252-02:00</from>
      <to>2013-11-18T18:38:56.942-02:00</to>
    </records>
    <records>
      <from>2013-11-18T18:39:08.321-02:00</from>
      <to>2013-11-18T18:39:10.080-02:00</to>
    </records>
  </day>
</days>
```

JSON:

```
[ {
  "id" : "2013-11-10T00:00:00.000-02:00",
  "records" : [ {
    "from" : "2013-11-10T18:44:01.579-02:00",
    "to" : "2013-11-10T18:44:09.717-02:00"
  } ]
}, {
  "id" : "2013-11-18T00:00:00.000-02:00",
  "records" : [ {
    "from" : "2013-11-18T18:38:30.252-02:00",
    "to" : "2013-11-18T18:38:56.942-02:00"
  }, {
    "from" : "2013-11-18T18:39:08.321-02:00",
    "to" : "2013-11-18T18:39:10.080-02:00"
  } ]
} ]
```

Получение записей за месяц (показывает, в какие дни сентября есть записи):

[http://example.com:\[port\]/somecontext/secure/archive/CAM:2/2011-12/](http://example.com:[port]/somecontext/secure/archive/CAM:2/2011-12/)

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<days>
  <day>
    <id>2011-09-02T00:00:00-05:00</id>
  </day>
  <day>
    <id>2011-09-03T00:00:00-05:00</id>
  </day>
  <day>
    <id>2011-09-05T00:00:00-05:00</id>
  </day>
</days>
```

JSON:

```
[ {
  "id" : "2011-09-01T00:00:00-0500",
  "records" : [ ]
}, {
  "id" : "2011-09-03T00:00:00-0500",
  "records" : [ ]
}, {
  "id" : "2011-09-01T00:00:00-0500",
  "records" : [ ]
} ]
```

Если записей нет, то присылается

XML:

<days/>

JSON:

[]

Нотификация

В разделе:

- Подписка на сообщения
- Аннулирование подписки
- Формат сообщения APN

Используются системы нотификации APNS(iOS), C2DN (Android) и т.д.

deviceid – device token (APNs), registration id (в случае C2DN) и т.д.;

username – логин пользователя. Может быть пустой.

Подписка на сообщения

Приложение при соединении с сервисом может осуществить подписку на сообщения APNS. В этом случае при выходе из программы на устройство будут приходить уведомления о тех или иных событиях.

POST

http://example.com:[port][somecontext]/secure/subscription/

Ответ с кодом "201 Created" означает, что подписка прошла успешно.

Код 400 означает, что параметры заданы не верно (deviceId не должно быть пустым, должно быть длиной от 5 до 150 символов и содержать только цифры и буквы английского алфавита).

Тело POST должно содержать информацию о создаваемой подписке. Принимается только формат JSON. Требуется корректно проставлять заголовок Content-Type.

Пример ответа:

JSON

Content-Type : application/json

```
{
  "username" : "johndoe",
  "deviceId" : "somedeviceid"
}
```

Аннулирование подписки

Аннулирование подписки происходит в следующих случаях:

- Пользователь подписался на события с другого устройства;
- Сменился device token или registration id;
- Другой пользователь подписался на события с данного устройства;
- Произошла ручная отписка от сообщений.

DELETE

http://example.com:[port]/[somecontext]/secure/subscription/[deviceId]

Ответ с кодом "204 No Content" означает, что подписка прошла успешно.

Формат сообщения APN

```
{
  "aps" : {
    "alert" : "Motion Detected",
    "badge" : 2 // .
  },
  "e" : {
    "srv" : "XXX", //id . ios
    "stt" : 88, //id (. )
    "obj" : "6", //id
  }
}
```

```
        "ts" : "2010-08-02T23:30:00Z" //
    }
}
```

Звук

Получение живого звука

GET

`http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionId=FC126734&command=audio.play&audio_in=MIC:5&format=L16&login=XXX&password=YYY`

sessionId – идентификатор сессии (тут пока не используется).

audio_in – идентификатор аудиопотока.

format – формат аудиоданных (пока только L16).

В ответ будут получены аудиопакеты в следующем виде:

HTTP/1.0 200 OK

Connection: close

Server: ITV-Intellect-Webserver/4.9.0.0

Cache-Control: no-store,no-cache,must-revalidate,max-age=0

Pragma: no-cache

Date: Mon, 13 Jan 2013 10:44:27 GMT

Content-Type: multipart/mixed;boundary=audioframe

--audioframe

Content-Type: text/xml

Content-Length: 138

<audio_in>

<sessionId>FC126734</sessionId>

<audio_in>MIC:5</audio_in>

<newstate>started</newstate>

<errcode>100</errcode>

```
</audio_in>
--audioframe
Content-Type: audio/L16;rate=8000;channels=1
Content-Length: 1024
X-Time: 2013-03-22T13:16:31.371+04:00
```

```
<audio packet PCM16>
--audioframe
Content-Type: audio/L16;rate=8000;channels=1
Content-Length: 1278
X-Time: 2013-03-22T13:16:31.873+04:00
<audio packet PCM16>
```

Для остановки потока без разрыва соединения необходимо в этом же соединении отправить команду

```
GET
http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionId=29101F1&command=audio.stop&audio_in=MIC:5&login=XXX&password=YYY
```

В этом случае в потоке придёт завершающий xml пакет:

```
--audioframe
Content-Type: text/xml
Content-Length: 106
<audio_in>
  <sessionId>FC126734</sessionId>
  <audio_in>MIC:5</audio_in>
  <newstate>closed</newstate>
  <errcode>100</errcode>
</audio_in>
```

Проигрывание звука из архива

```
GET
http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionId=29101F1&command=arc.play&audio_in=MIC:5&format=L16&time_from=2013-03-22T13:16:31.873+04:00&time_to=2013-03-22T13:04:52.312+04:00&login=XXX&password=YYY
```

audio_in – идентификатор аудиопотока;

format – запрашиваемый формат (сейчас пока только L16);

time_from - время начала проигрывания архива;

time_to - время завершения проигрывания архива.

Поток приходит в том же виде, что и при живом звуке.

При завершении данных приходит завершающий xml пакет (как при получении живого звука – см. [Получение живого звука](#)).

Отправка живого звука

Отправка звука идёт последовательной передачей пакетов командами:

POST

`http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionId=FC126734&command=audio.receive&audio_out=SPEAKER:3&login=XXX&password=YYY`

`Content-type: audio/L16;rate=8000;channels=1`

`Connection: keep-alive`

Далее происходит передача аудиопакета.

Формат звука – только L16.

rate – любое разумное значение.

channels – от 1 до 6.