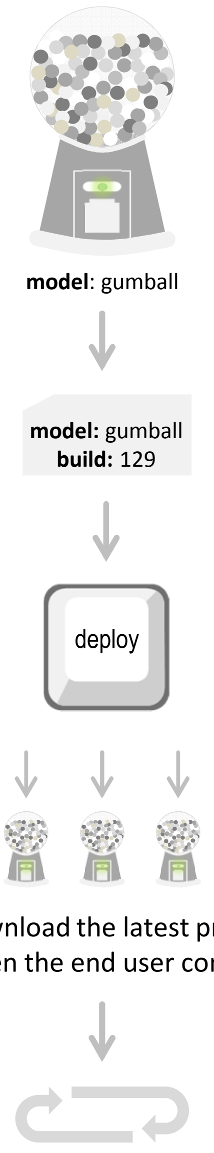
electric imp: Blessing
module

july 2013
www.electricimp.com

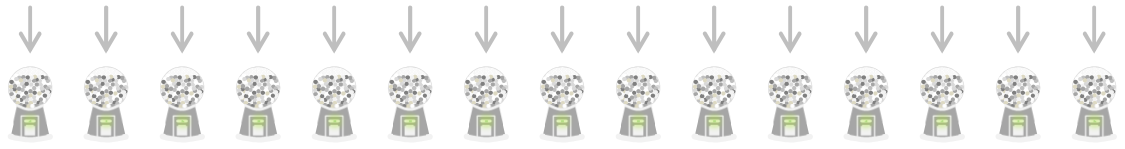
Blessing in context



Develop model firmware in the IDE

Promote a “gumball” build to Ops Console

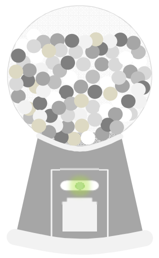
Confirm and deploy chosen build



Blessed devices automatically download the latest production firmware of its model – model: gumball, build: 129 – when the end user connects the device to the Internet

Repeat “develop > promote > deploy” process any time

What you need to bless devices



model: gumball

- Upon blessing, the device's unique ID is assigned to a model e.g., model: gumball, in the Electric Imp database. When the end user connects to the device to the Internet it will run the latest deployed build of that model

For blessing devices to a model, you will need:

Model factory firmware

- Factory firmware runs on each device on the production line to test and bless the device.
- The same factory firmware also runs on imps in factory LED fixtures.

Factory imp(s)

- A factory imp is used in factory LED fixtures, whose purpose is to send a special *factory BlinkUp* token to the imp modules within production devices. This token instructs the modules in the products to download and run factory firmware.
- As many factory imps and LED fixtures as required can be used.

Factory LED fixture

- The factory test fixture consists of LED(s) which send the *factory BlinkUp* tokens, under control of the factory imp.
- A single factory imp can drive an LED array to instruct multiple devices to run factory firmware concurrently.
- The *factory BlinkUp* process runs at over 2x the speed of consumer-level BlinkUp.

Production line

- To bless devices on the production line, the factory imp and modules in the devices will need access to the Internet via WiFi and each device will need to be powered on during *factory BlinkUp* and blessing.

HTTP callbacks*

- Webhooks can be used to track device test results and blessing.

* Optional

Quick overview of how-to bless

1 **Develop factory firmware**

- The factory firmware implements two distinct functions, one of which is selected at runtime based on which device it is running on (usually, card or module):
  - When on a card in the factory LED fixture, it sends *factory BlinkUp* tokens.
  - When on a module, it tests the device and, upon pass, blesses it.
- You will need to use the bless and *factory BlinkUp* APIs in the factory firmware

2 **Designate factory imp(s) and create factory test fixture**

- The factory test fixture requires a factory imp with an LED (to send the blinkup tokens), button (to initiate the transmission). This can be designed to meet your specific factory needs.
- A minimum of 1 factory test fixture with a factory imp is needed.
- The factory imp in the factory test fixture only needs to be BlinkedUp once via the Electric Imp app

3 ***Factory BlinkUp* and run factory firmware**

- Align the factory test fixture LED with the device's phototransistor and press the button on the factory test fixture to start *factory BlinkUp*. Wait for *factory BlinkUp* to finish and then the imp module in the device will download and run the factory firmware; after bless or test fail, indicated by solid green or red LED on the device, use the factory test fixture with the next device(s).

4 **View model activity** (on-going)

- Track the number of blessed devices per model in the Ops Console*

5 **Ship devices; end-user BlinkUp**

- When the end user connects the device to the Internet using BlinkUp in your app, it will automatically download and run the model firmware

* Currently in development

1 Develop factory firmware

The factory firmware is responsible for testing and blessing your new devices. It's run on the new devices when a factory imp in an LED fixture sends a factory BlinkUp token to the new device.

Factory firmware should perform any desired diagnostic tests on the device's hardware – checking peripherals can be accessed, and so on. After a successful test, it blesses the device via the blessing API, which permanently associates the unique ID of the device with your model ID – and hence ensures it will automatically load the correct firmware from then onwards.

Develop your factory firmware using the IDE.

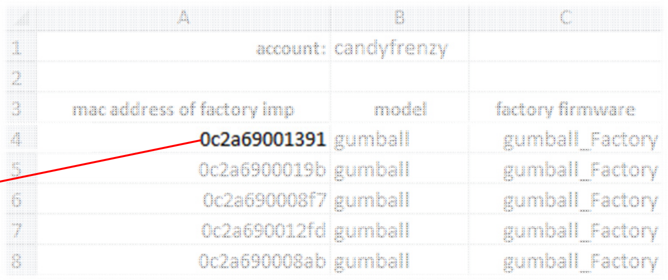
Factory firmware		
server.factoryblinkup **(ssid, pw, pin, flags)** *API call from factory test fixture*	**test** *"customer defined"* [firmware only]	**server.bless** **(testPass, callback)** *API call from factory firmware*
Sends a special *factory BlinkUp* token from the factory imp in the factory test fixture to one or more imp module(s) in device(s). The *factory BlinkUp* token instructs the module(s) in the device(s) to download and run factory firmware for testing and blessing the device(s). *Factory BlinkUp* is transmitted from the factory test fixture LED(s) to the device(s) phototransistor.	Runs diagnostic test on your device's hardware. If testPass is true, the device is blessed and the LED connected to the imp module is turned solid green for success. If testPass is false, the device is *not* blessed and the LED connected to the imp module is turned solid red for failure.	If testPass is true, permanently enrolls a devices as part of a specified model so that when the end-user connects the device via BlinkUp the device will download the latest deployed model build from your account. The LED is turned on solid green and the callback is called after the blessing has been completed. If testPass is false, turns the imp LED on solid red. No blessing is performed and the callback is not called.

Notes:
- Develop your factory firmware using the IDE; see details on the API calls in the appendix.

2 Designate factory imp(s) and create factory test fixture

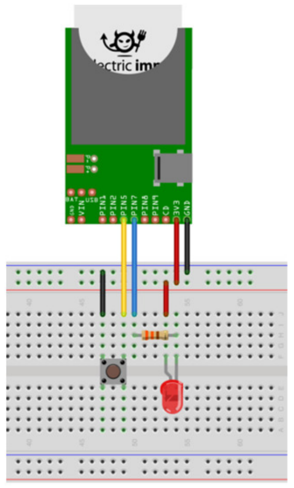
Factory imps are required for the factory test fixture and will run factory firmware to ensure it sends the correct factory BlinkUp *token to the device's phototransistor. Contact Electric Imp with your account name, model that you want devices to be blessed to, the factory firmware name, and mac address(es) of the imp card(s) that you want to designate as factory imp(s).*

factory
electric imp
designed in california
mac:0c2a69001391
CE FC



	A	B	C
1	account:	candyfrenzy	
2			
3	mac address of factory imp	model	factory firmware
4	0c2a69001391	gumball	gumball_Factory
5	0c2a6900019b	gumball	gumball_Factory
6	0c2a690008f7	gumball	gumball_Factory
7	0c2a690012fd	gumball	gumball_Factory
8	0c2a690008ab	gumball	gumball_Factory

factory imp

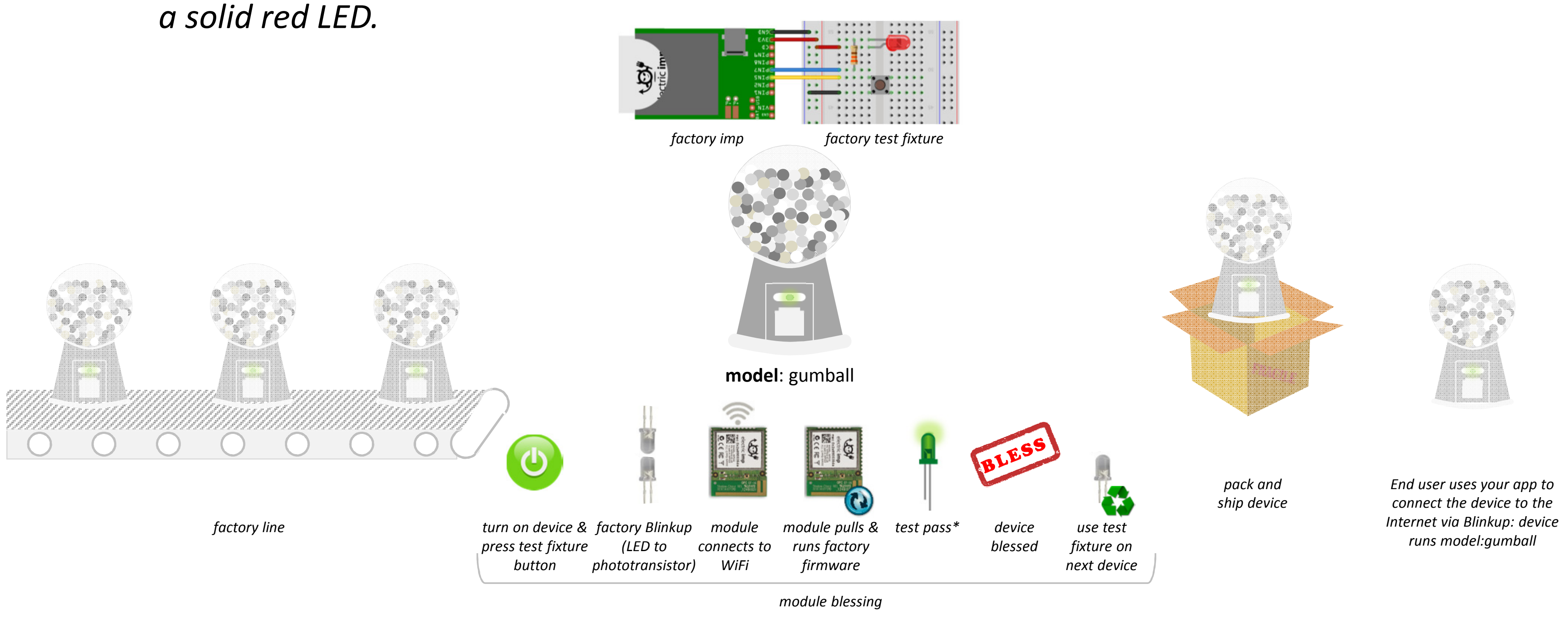


example factory test fixture

Notes:
- Any imp card can be used as a factory imp
- You should mark the card so you do not switch it with another card accidentally
- Just one factory imp and factory test fixture is required for each production line but you may designate, create and use as many as you like
    - One factory test fixture can factory BlinkUp multiple modules at one time if the LED on the factory fixture is bright enough
- Factory imps can be re-designated to a new model and new factory firmware and used again after a production run
- To connect your factory imp(s) to the Internet, insert it into the factory test fixture and power on the factory test fixture then use BlinkUp via the Electric Imp app. This needs to be done only once; the factory imp will continue to use the network it is connected to automatically.

3 *Factory BlinkUp* and run factory firmware

Insert the factory imp into a factory test fixture and connect it to the Internet with the Electric Imp app. For devices on the production line, align the LED on the test fixture with the phototransistor connected to the module and press the button on the factory test fixture to send factory BlinkUp. *Afterwards, the module will ping the electric imp server and receive and run its model's factory firmware. If the device passes the test(s) then the device will be blessed and indicate pass by turning the attached LED on solid green. A fail is indicated with a solid red LED.*

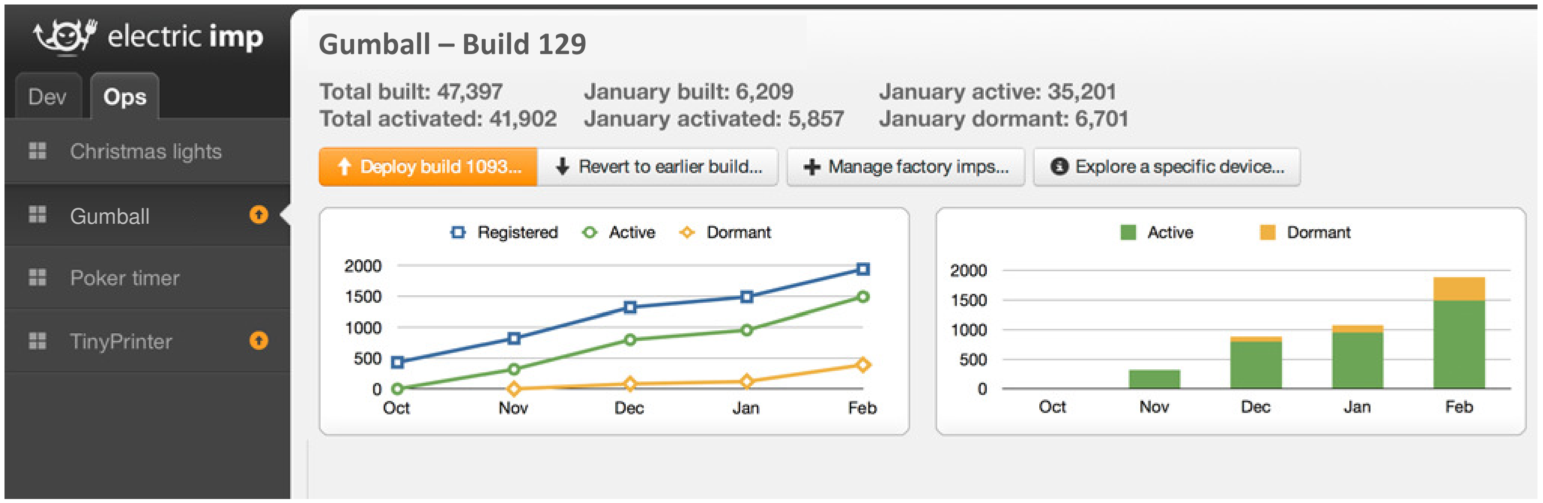

factory imp
factory test fixture
model: gumball
factory line
turn on device & press test fixture button
factory Blinkup (LED to phototransistor)
module connects to WiFi
module pulls & runs factory firmware
test pass*
BLESS
device blessed
use test fixture on next device
module blessing
pack and ship device
End user uses your app to connect the device to the Internet via Blinkup: device runs model:gumball

* The LED connected to the imp module will turn red if it does not pass the test

4

View model activity

To track blessed devices, you can view the Ops Console. This tool allows you to view the number of devices that are built, activated, active and dormant. Additionally, the Ops Console allows you to deploy new model builds if you have firmware updates.*



Gumball – Build 129

Total built: 47,397 January built: 6,209 January active: 35,201
Total activated: 41,902 January activated: 5,857 January dormant: 6,701

[built] = the number of devices that were successfully blessed on the production line via a factory imp

[registered/activated] = the number of devices as they are connected by the end user via BlinkUp

[active] = the number of devices that have connected to the Electric Imp Services in a calendar month

[dormant] = the number of devices that have not connected to the Electric Imp Services in a calendar month

Notes:

- If you want to use webhooks you will be able to add them in the IDE in the “Model Settings”. Until this feature is added, contact Electric Imp to set up a wehbook to a specified model.

* Currently in development

5 End-user BlinkUp

The device will automatically download the latest deployed build of its model firmware when the end user connects the device to the Internet with BlinkUp in your app.

end user + your app

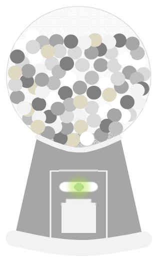
backend

1 User enters credentials and/or data into your app e.g., username, email, location, preferences, etc.*

- App provides an API key** to the BlinkUp SDK to identify and verify the account and then waits for the agent URL
- Electric Imp server generates a setup token to verify the account/device and plan ID and sends this back to the SDK

2 User enters WiFi credentials into your app and starts BlinkUp

- The SDK sends the WiFi details and setup token to the imp via BlinkUp
- The imp sends Electric Imp the device ID, setup token and plan ID
- Electric Imp notifies your server that the device completed registration (optional)
- Electric Imp starts new agent VM and returns the agent URL to your app
- Electric Imp sends the latest build to the imp in the device
- App sends data to the agent

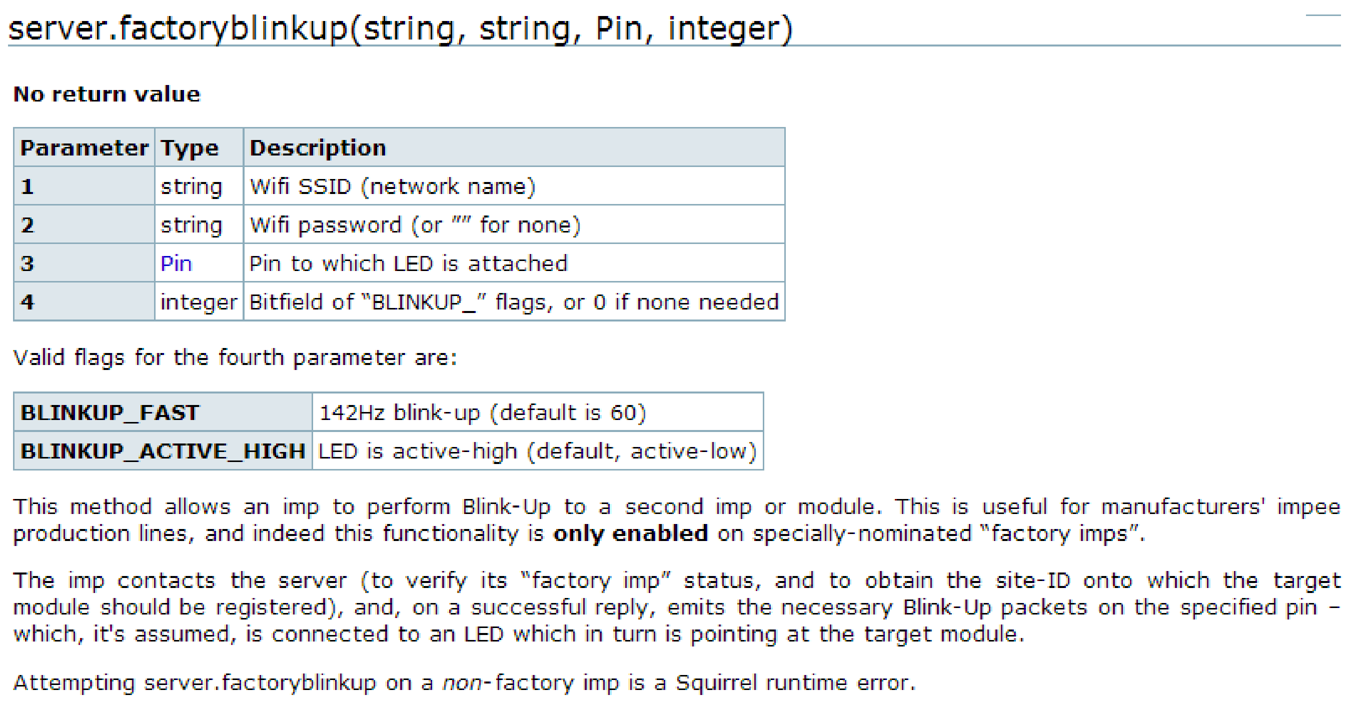


end-user can start using the device's connectivity features and services

* personally identifiable information (PII) stored on your server(s); PII will not be stored on Electric Imp servers
** Electric Imp will provide an API key when the SDK is sent to you that will be associated to your Electric Imp account

appendix

server.factoryblinkup



server.factoryblinkup(string, string, Pin, integer)

No return value

Parameter	Type	Description
1	string	Wifi SSID (network name)
2	string	Wifi password (or "" for none)
3	Pin	Pin to which LED is attached
4	integer	Bitfield of "BLINKUP_" flags, or 0 if none needed

Valid flags for the fourth parameter are:

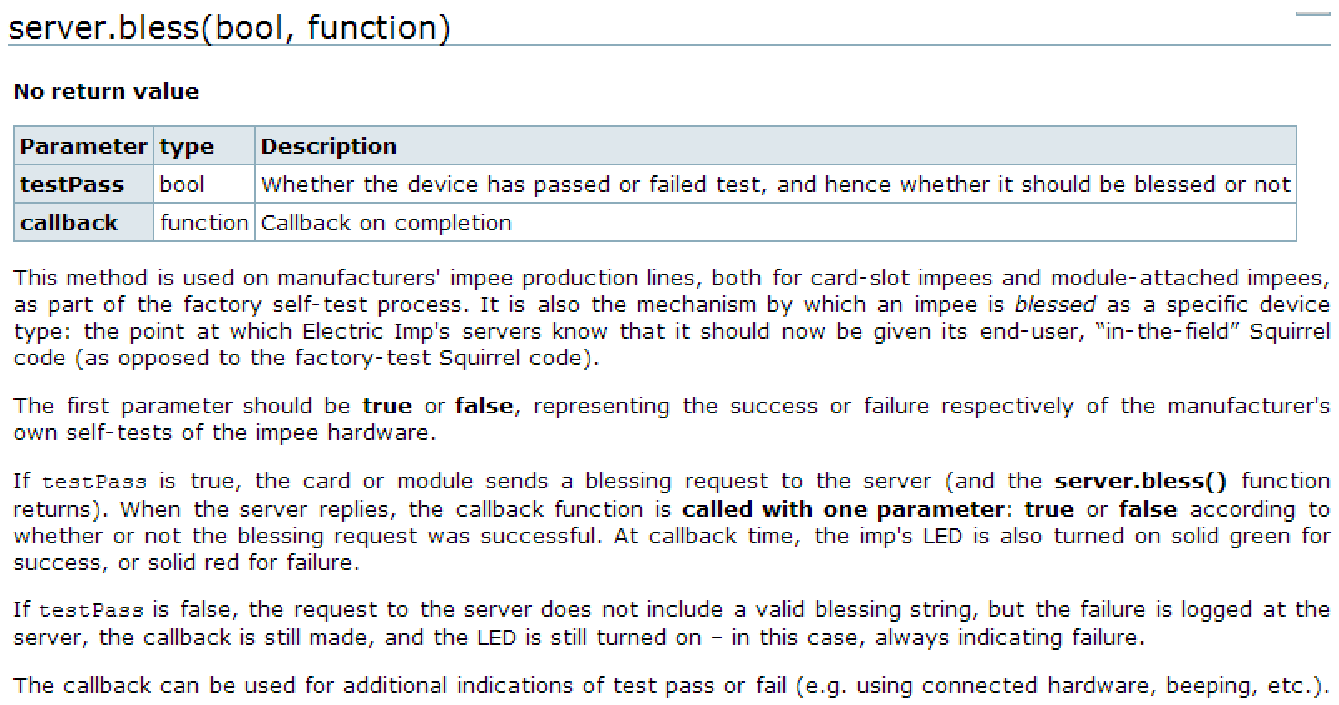
BLINKUP_FAST	142Hz blink-up (default is 60)
BLINKUP_ACTIVE_HIGH	LED is active-high (default, active-low)

This method allows an imp to perform Blink-Up to a second imp or module. This is useful for manufacturers' impee production lines, and indeed this functionality is **only enabled** on specially-nominated "factory imps".

The imp contacts the server (to verify its "factory imp" status, and to obtain the site-ID onto which the target module should be registered), and, on a successful reply, emits the necessary Blink-Up packets on the specified pin – which, it's assumed, is connected to an LED which in turn is pointing at the target module.

Attempting server.factoryblinkup on a *non*-factory imp is a Squirrel runtime error.

server.bless



server.bless(bool, function)

No return value

Parameter	type	Description
testPass	bool	Whether the device has passed or failed test, and hence whether it should be blessed or not
callback	function	Callback on completion

This method is used on manufacturers' impee production lines, both for card-slot impees and module-attached impees, as part of the factory self-test process. It is also the mechanism by which an impee is *blessed* as a specific device type: the point at which Electric Imp's servers know that it should now be given its end-user, "in-the-field" Squirrel code (as opposed to the factory-test Squirrel code).

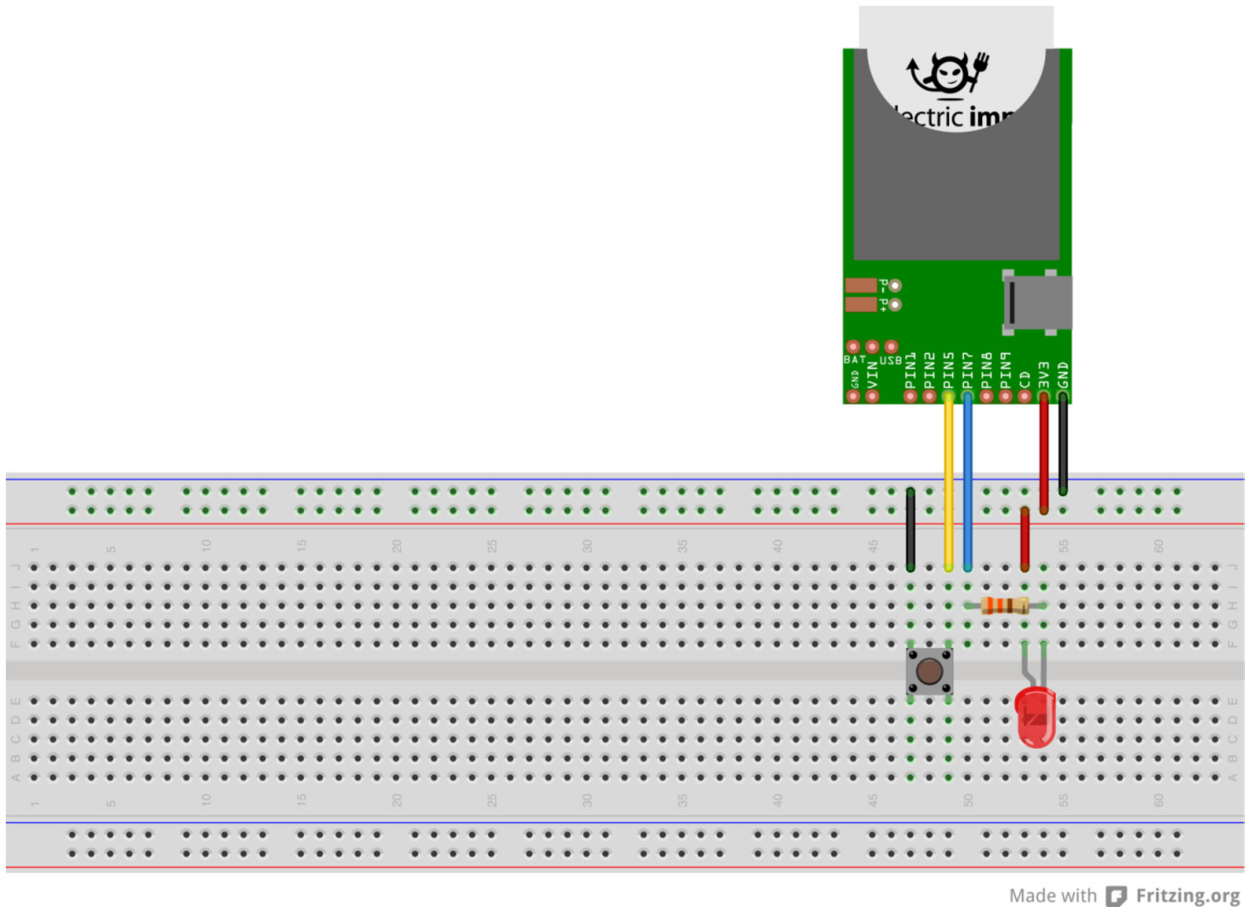
The first parameter should be **true** or **false**, representing the success or failure respectively of the manufacturer's own self-tests of the impee hardware.

If `testPass` is true, the card or module sends a blessing request to the server (and the **server.bless()** function returns). When the server replies, the callback function is **called with one parameter**: **true** or **false** according to whether or not the blessing request was successful. At callback time, the imp's LED is also turned on solid green for success, or solid red for failure.

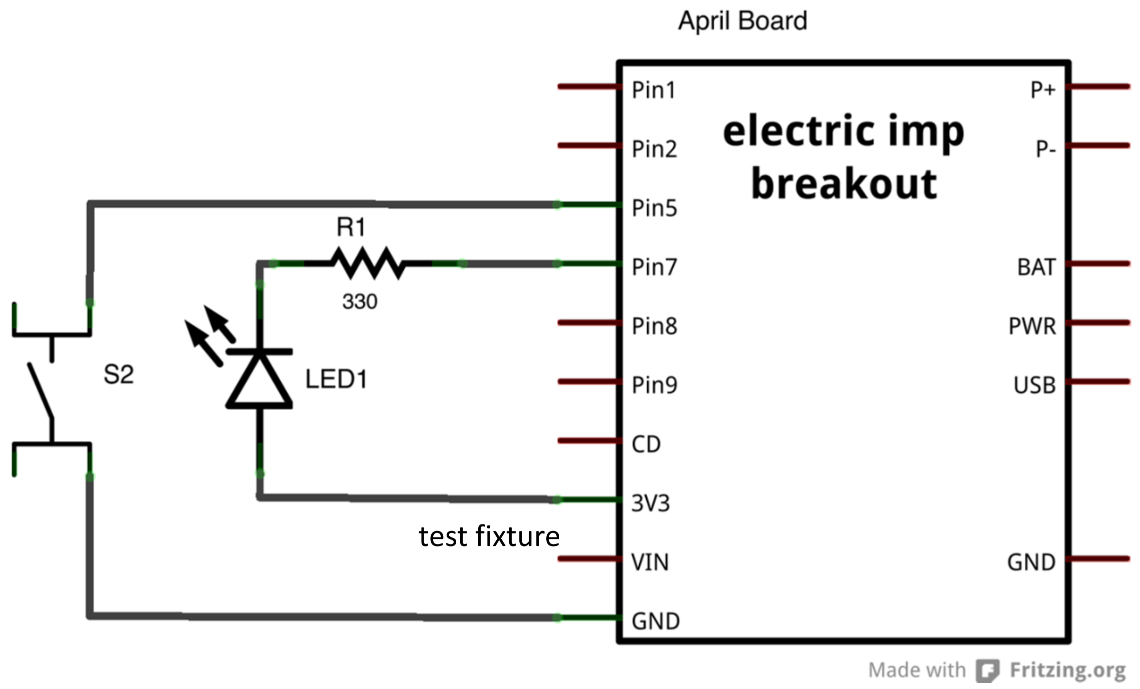
If `testPass` is false, the request to the server does not include a valid blessing string, but the failure is logged at the server, the callback is still made, and the LED is still turned on – in this case, always indicating failure.

The callback can be used for additional indications of test pass or fail (e.g. using connected hardware, beeping, etc.).

Factory test fixture: breadboard example



Factory test fixture: simple schematic


April Board
electric imp breakout
Pin1
Pin2
Pin5
Pin7
Pin8
Pin9
CD
3V3
VIN
GND
P+
P-
BAT
PWR
USB
GND
R1
330
S2
LED1
test fixture
Made with Fritzing.org