# Structure & Protocol Element Definitions for Jacket JSON

**Version 0.4**

The first part of this document describes the JSON structures for the jacket’s plans and phases. The second part describes the protocol elements that are exchanged between an app and the jacket (and vice versa).

The following general rules and restrictions apply:

* All field names and values are case significant.
* The available size for a single plan is currently 512 bytes.
* All JSON data shall be compressed when sent via BLE, ie. without any formatting.
* Every plan and phase start with all LED switched off.

## Plan & Phases JSON Structures

### Plan

The schema of a training plan:

{  
 "na": <string>,  
 "ph": [ <list of Phase> ],  
 "it": <integer>  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| na | String | Yes | empty string | The name of the plan. |
| ph | List of Phase descriptions | No | - | Ordered list of phases (sb). |
| it | Integer | Yes | 1 | Number of repetitions of a plan. 0 = forever |

### Phase

The schema of a single phase definition:

{  
 "na":<string>,  
 "co":<color>,  
 "bd":<blink direction>,  
 "du":<long>,  
 "bi":<long>,  
 "in":<long>  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| na | String | Yes | empty string | The name of the phase. |
| co | String, color definition | Yes | “n” | The name of a color (sb). |
| bd | String, blink direction | Yes | “f” | The name of a blink direction (sb). |
| du | Long integer | No | - | Duration of the phase in ms. |
| bi | Long integer | Yes | value of *du* | Blink interval in ms. |
| in | Long integer | Yes | - | Intensity, e.g. for Trainings. The jacket will ignore this field. |

### Color Definition

The following abbreviated values define color values:

|  |  |
| --- | --- |
| Value | Color |
| r | Red |
| y | Yellow |
| g | Green |
| b | Blue |
| n | No color, off |

### Blink Definition

The following abbreviated values define blink behaviours:

|  |  |
| --- | --- |
| Value | Blink Meaning |
| f | Forward |
| r | Reverse, backward |
| s | Static |
| b | Blink |

**TODO** The options of this table need to be aligned with the use cases.

### Example

The following example shows a plan and its phases.

* Some of the optional attributes, such as *plan.na*, *phase[0].na*, *plan.it* etc are provided to show their use.
* The last phase expands to “color: none, bs: 5000, bd:f”.

{  
 "na":"Testplan",  
 "ph":[  
 {  
 "na":"red",  
 "co":"r",  
 "bd":"f",  
 "bi":500,  
 "du":5000  
 },  
 {  
 "co":"y",  
 "bi":200,  
 "du":5000  
 },  
 {  
 "co":"g",  
 "bi":100,  
 "du":5000  
 },  
 {  
 "co":"b",  
 "bi":50,  
 "du":5000  
 },  
 {  
 "du":5000  
 }  
 ],  
 "it":0  
}

## Protocol Element Definitions

The definitions in this section specify the protocol elements for the BLE communication between an app and a jacket. Every command and answer is sent in a single BLE datagram.

**Commands Overview**

|  |  |
| --- | --- |
| Command | Description |
| npl | [Load Plan into Jacket](#load-plan-into-jacket-app-jacket) |
| spl | [Start Plan in Jacket](#start-plan-in-jacket-app-jacket) |
| sph | [Load and Start a Phase](#load-and-start-a-phase-app-jacket) |
| stp | [Stop a Plan](#stop-a-plan-app-jacket) |
| pse | [Pause a Plan](#pause-a-plan-app-jacket) |
| prv | [Start Previous Phase](#start-previous-phase-app-jacket) |
| nxt | [Start Next Phase](#start-next-phase-app-jacket) |
| rst | [Reset the Jacket](#reset-the-jacket-app-jacket) |

**Signals Overview**

|  |  |
| --- | --- |
| Signal | Description |
| eopl | [Signal End of Plan](#signal-end-of-plan-jacket-app) |
| eoph | [Signal End of Phase](#signal-end-of-phase-jacket-app) |
| btn | [Signal Button Press](#signal-button-press-jacket-app) |
| bat | [Signal Battery Level](#signal-battery-level-jacket-app) |

### Load Plan into Jacket (App → Jacket)

This JSON structure defines the protocol element to load a plan into a jacket. The plan is not started. It is send from an application to the jacket.

If successfully transmitted and processed by the jacket, the plan replaces an exsisting plan in the jacket.

If the jacket is currently executing a plan, that plan is stopped.

{  
 "co": "npl",  
 "pl": <plan>  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| co | String | No | - | The command. Must be “npl”. |
| pl | JSON Structure | No | - | A plan as defined above. |

If successful the jacket MUST answer with a positive acknowledge, otherwise with a negative response (s.b.).

### Start Plan in Jacket (App → Jacket)

This JSON structure defines the protocol element to start a previously loaded plan into a jacket. This command, when given without a plan description, also unpauses a previously paused (not stopped!) plan ([Pause a Plan](#pause-a-plan-app-jacket)).

Optionally, a plan can be transmitted together with this command. See the description above ([Load Plan into Jacket](#load-plan-into-jacket)).

{  
 "co": "spl",  
 "pl": <plan>  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| co | String | No | - | The command. Must be “spl”. |
| pl | JSON Structure | Yes | - | A plan as defined above. |

If successful the jacket MUST answer with a positive acknowledge, otherwise with a negative response (s.b.).

### Load and Start a Phase (App → Jacket)

This JSON structure defines the protocol element to load into a jacket and start it. The jacket handles the loading of a phase as a plan with a single phase and with no repetition.

If successfully transmitted and processed by the jacket, the phase replaces an exsisting running phase in the jacket.

If the jacket is currently executing a phase, that phase is stopped.

{  
 "co": "sph",  
 "pl": <phase>  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| co | String | No | - | The command. Must be “sph”. |
| ph | JSON Structure | No | - | A single phase as defined above. |

If successful the jacket MUST answer with a positive acknowledge, otherwise with a negative response (s.b.).

### Stop a Plan (App → Jacket)

This JSON structure defines the protocol element to stop a running plan. When started again (see [Start Plan in Jacket](#start-plan-in-jacket-app-jacket)) the plan starts from the beginning.

{  
 "co": "stp"  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| co | String | No | - | The command. Must be “stp”. |

If successful the jacket MUST answer with a positive acknowledge, otherwise with a negative response (s.b.).

### Pause a Plan (App → Jacket)

This JSON structure defines the protocol element to pause a running plan. When started again (see [Start Plan in Jacket](#start-plan-in-jacket-app-jacket)) the plan starts where it was paused.

{  
 "co": "pse"  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| co | String | No | - | The command. Must be “pse”. |

If successful the jacket MUST answer with a positive acknowledge, otherwise with a negative response (s.b.).

### Start Previous Phase (App → Jacket)

This JSON structure defines the protocol element to stop the current and activate the previous phase. If there is no previous phase then this command is ignored.

{  
 "co": "prv"  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| co | String | No | - | The command. Must be “prv”. |

If successful the jacket MUST answer with a positive acknowledge, otherwise with a negative response (s.b.).

### Start Next Phase (App → Jacket)

This JSON structure defines the protocol element to stop the current and activate the next phase. If there is no next phase then this command is ignored.

{  
 "co": "nxt"  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| co | String | No | - | The command. Must be “nxt”. |

If successful the jacket MUST answer with a positive acknowledge, otherwise with a negative response (s.b.).

### Reset the Jacket (App → Jacket)

This JSON structure defines the protocol element to reset a jacket to an initial state. This includes stopping and removing all phases and plans, setting the LEDs to off-state, etc.

{  
 "co": "rst"  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| co | String | No | - | The command. Must be “rst”. |

If successful the jacket MUST answer with a positive acknowledge, otherwise with a negative response (s.b.).

### Signal End of Plan (Jacket → App)

This JSON structure defines the protocol element to signal the end of a plan from the jacket to an app.

{  
 "si": "eopl"  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| si | String | No | - | The signal. Must be “eopl”. |

Sending this protocol element DOES NOT require an answer from the receiving app.

### Signal End of Phase (Jacket → App)

This JSON structure defines the protocol element to signal the end of a phase from the jacket to an app.

{  
 "si": "eoph"  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| si | String | No | - | The signal. Must be “eoph”. |

Sending this protocol element DOES NOT require an answer from the receiving app.

### Signal Button Press (Jacket → App)

This JSON structure defines the protocol element to signal the press of a button / a button event from the jacket to an app.

{  
 "si": "btn",  
 "bt": <button type>  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| si | String | No | - | The signal. Must be “btn”. |
| bt | String | No | - | The button event type (sb). |

Sending this protocol element DOES NOT require an answer from the receiving app.

#### Button Types

The following abbreviated values define button event types:

|  |  |
| --- | --- |
| Value | Button Event |
| a | Button A touched only. |
| b | Button B touched only. |
| ab | Both button A and B touched simultaneously. |
| a2b | Slide-touch from button A to B. |
| b2a | Slide-touch from button B to A. |
| ice | Emergency touch gesture. |

### Signal Battery Level (Jacket → App)

This JSON structure defines the protocol element to signal the battery level of the jackets internal battery to an app.

{  
 "si": "bat",  
 "bl": <integer>  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| si | String | No | - | The signal. Must be “bat”. |
| bl | Integer | No | - | The current battery level of the jacket. |

Sending this protocol element DOES NOT require an answer from the receiving app.

**NOTE**: We need to define what the most meaningful battery level would be. Charge? %? time remaining?

### Answers

**Positive Acknowledgment**

A positive acknowledge is sent as an answer to a message when the message

* was received, and
* was correctly parsed, and
* was correctly executed.

**Negative Acknowledgment**

A negative acknowledge is sent as an answer when any of above conditions was not met. In this case a reason MUST be given to the sender. See the status codes below.

**Acknowledgment Structure**

{  
 "ack": <bool>,  
 "rs": <integer>  
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Type | Optional | Default | Description |
| ack | Boolean | No | - | True or false, depending on the result. |
| rs | Integer | Yes | 200 | A status code, similar to http status codes. |

#### Status Codes

|  |  |
| --- | --- |
| Status Code | Description |
| 200 | OK. Positive acknowledgment. |
| 204 | No Content. If the message was empty. |
| 400 | Bad Request. If the JSON was incorrect. |
| 405 | Method Not Allowed. Method not allowed at this time. |
| 500 | Internal Server Error. Should not happen. |
| 501 | Not Implemented. Method not implemented yes. |

**NOTE**: Anything else?

## Future Extensions

* Feature: Instead of defining a phase that does the same as a previous one, reference that defined phase {"re":"nameOfPhase"}

## Changes

#### Version 0.4

* Fixed typos.
* Fixed battery level type.
* Added *nxt* and *prv* commands.

#### Version 0.3

* Fixed wrong *stp* command in command overview table.
* Added version history section.
* Added page numbers for the Word doc.

#### Version 0.2

* Specified protocol elements.

#### Version 0.1

* Initial version.