**DVP Elektroinstallation JSON formate Version 1.5 12.7.2018**

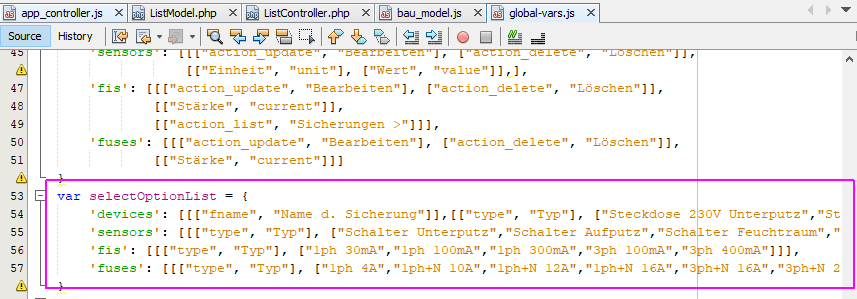
History:

1. Alle CUD und List definitionen, sowie alle “specifications“
   1. plus die definitionen für Shoppinglist und Sicherungskreisliste
   2. ….id namen vereinheitlicht; Aufbau Auswertungslisten ergänzt (alle Änderungen/Erweiterungen/Abweichungen vom FLOformat im text in ROT)
   3. Feld für Bezeichnung („name“ ) in den Update und Create formaten ergänzt
   4. fuseid statt sicherungid
   5. circuitlist um fuseid ergänzt, damit flo diese für die dropdownlist (sicherungswahl) im device verwnden kann; type’s in devices, sensors, fis, fuses ergänzt .. alles in GRÜN

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Kommunikationszweck*** | ***Request (POST)*** | ***Reply (Format: FLO)*** |
| **L** | **Anforderung der Liste --> "action":"list"** |  | !!! Inhalt von „specification“  Siehe eigene Tabelle „SPECIFICATIONS“ (***S1 – S7***) |
| L1 | aller Projekte | { "data":  {"action":"list","listtype":"projects"}  } | {ProjektObjekt =  {  "project-1": {          "titel": "Under the Bridge",  "specification": {***S1***}      } ,      "project-2": {          "titel": "Todesstern",          "specification": {***S1***}      } , ……  }  } |
| L2 | aller "floors" des Projektes mit der parentID: X (zB: 1) | {"data":  {"action":"list","listtype":"floors","parentid":1}  } | {FloorObjekt = {      "floor-1": {          "titel": "Kerker",  "parentid": 4,  "specification": {***S2***}      } ,      "floor-2": {          "titel": "Audienz-Ebene",  "parentid": 4,          "specification": {***S2***}      }, …….  }  } |
| L3 | aller "rooms" des floors mit der parentID: X (zB: 1) | detto mit listtype: "rooms" | {RoomObjekt = {      "room-1": {          "titel": "Zelle #666",  "parentid": 1,  "specification": {***S3***}  } , …..  } |
| L4 | detto für devices (parentid = floor.id) | detto mit listtype: "devices" | {DeviceObjekt = {      "device-1": {          "titel": "Elektro-Fackel",  "parentid": 3,          "specification": {***S4***}      }, ……….  }  } |
| L5 | detto für sensors (Die wir als SCHALTER bezeichnen) | detto mit listtype: "sensors" | {SensorObjekt = {      "sensor-1": {          "titel": "Lichtschalter",  "parentid": 1,          "specification": {***S5***}      }, …….  }  } |
| L6 | aller "FI's" des Projektes mit der parentID: X (zB: 1) | { "data":{"action":"list","listtype":"fis","parentid":1}} | {FiObjekt = {  "fi-1": {  "titel": "FI #1",  "parentid": 1,          "specification": {***S6***}  }, ……  }  } |
| L7 | detto für Sicherungen (parentID = fi.id) | detto mit listtype: "fuses" | {FuseObjekt = {  "fuse-1": {  "titel": "Sicherung #1",  "parentid": 1,          "specification": {***S7***}  }, ……  }  } |

# Selections / DropDown-Elemente

Um den Inhalt der DropDown-Elemente, für z.B. die Einstellung des Verbrauchertyps, aus der Datenbank zu lesen, werden neue POST-requests benötigt. Mit Hilfe dieser Requests kann dann die globale Variable selectOptionList gefüllt oder aktualisiert werden.



Es bietet sich an, die Daten für selectOpionList schon als Array an das Frontend zu senden. Falls das für Flo nicht passt, müssten wir halt wieder die Spec anpassen.

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Kommunikationszweck*** | ***Request (POST)*** | ***Reply (Format: FLO)*** |
| **L** | **Anforderung der Liste --> "action":"list"** |  |  |
| L4.1 | Verbrauchertypen (devicetypes) | { "data":  {"action":"list","listtype":"devicetypes"}  } | {  "devicetypes": [  {  "id": 1,  "name": „Steckdose 230V Unterputz",  }  ]  } |
| L5.1 | Sensortypen (sensortypes) | { "data":  {"action":"list","listtype":" sensortypes "}  } | {  " sensortypes ": [  {  "id": 1,  "name": „Schalter Unterputz",  }  ]  } |
| L6.1 | FI-Typen (fitypes) | { "data":  {"action":"list","listtype":" fitypes "}  } | {  " fitypes ": [  {  "id": 1,  "name": „1ph 30 mA",  }  ]  } |
| L7.1 | Sicherungstypen (fusetypes) | { "data":  {"action":"list","listtype":" fusetypes "}  } | {  " fusetypes ": [  {  "id": 1,  "name": „1ph 4A",  }  ]  } |

***Gegenüberstellung Reply Uebung4 / Reply Flo***

|  |  |  |
| --- | --- | --- |
| L2 | {  "listtypeNice": "Projekte",  "listtype": "FLOORS",  "items": [{  "id": "1",  "name": "Haus Großeltern",  "url": <http://localhost/Uebung3/index.php?listtype=FLOORS&parentid=1>  }, {  "id": "2",  "name": "Gebäude 1200 Wien",  "url": <http://localhost/Uebung3/index.php?listtype=FLOORS&parentid=2>  }  ParentID aus dem Request  ]  } | FloorObjekt = {      "floor-1": {          "titel": "Kerker",  "parent\_id": 4,          "specification": {              "countFromBasement": "-32",          }      },      "floor-2": {          "titel": "Audienz-Ebene",  "parent\_id": 4,          "specification": {              "countFromBasement": "3",          }      }  } |

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Kommunikationszweck*** | ***Request (POST)*** | ***Reply (Format: NEU)*** |
| **D** | **Delete eines Objektes --> "action":"delete"** |  |  |
| D1 | Delete EINES Projektes | { "data":  {"action":"delete","listtype":"projects",”itemid”:1}  } | {“status”:0 , “message”:”Failed to delete Item”}  {“status”:1 , “message”:”Item deleted”}  Anmerkung: “itemid“ ist die Nummer des objektes ,  also z.B für project-1 die nummer 1 (ohne “project“) |
| Dx | Delete generell | Detto für alle sonstigen “listtypes” | {“status”:0 , “message”:”Failed to delete Item”}  {“status”:1 , “message”:”Item deleted”}  Anmerkung: “itemid“ ist die Nummer des objektes ,  also z.B für floor-10 die nummer 10 (ohne “floor“) , etc… |

ACHTUNG: Um zu verhindern, daß Waisen in der Datenbank zurückbleiben, sollten wir versuchen die Tabellen in de DB mit constraints anzulegen (wenn das in MySQL überhaupt geht).

zB: Ein Projekt wird gelöscht, obwohl es darunter floors, rooms, devices etc. gibt …. Diese würden dann unansprechbar als Leichen in der DB zurückbleiben, da sie ja an der parentID des gelöschten Projektes hängen.

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Kommunikationszweck*** | ***Request (POST)*** | ***Reply (Format: NEU)*** |
| **U** | **Update eines Objektes --> "action":"update"** |  |  |
| U1 | Update EINES Projektes | { "data":  {"action":"update",  "listtype":"projects",  ”itemid”:1,  "specification": {  “name”: “Das Schöne Haus”,             "baumeister": "Harri Hurtig“,              "kapital": 1000  }  }  } | {“status”:0 , “message”:”Failed to update Item”}  {“status”:1 , “message”:”Item <itemid> updated”}  Anmerkung: “itemid“ ist die Nummer des objektes ,  also z.B für project-1 die nummer 1 (ohne “project“) |
| Ux | Update generell | Detto für alle sonstigen “listtypes” mit den entsprechenden “specifications” | {“status”:0 , “message”:”Failed to update Item”}  {“status”:1 , “message”:”Item <itemid> updated”}  Anmerkung: “itemid“ ist die Nummer des objektes ,  also z.B für floor-10 die nummer 10 (ohne “floor“) , etc… |

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Kommunikationszweck*** | ***Request (POST)*** | ***Reply (Format: NEU)*** |
| **C** | **Create eines Objektes --> "action":"create"** |  |  |
| C1 | Create eines NEUEN Projektes | { "data":  {"action":"create",  "listtype":"projects",  "parentid": 1,  "specification": {  “name”: “Das Schöne Haus”,              "baumeister": "Harri Hurtig“,              "kapital": 1000  }  }  } | {“status”:0 , “message”:”Failed to update Item”}  {“status”:1 , “message”:”Item <item\_id> updated”}  Anmerkung: “parentid“ ist die Nummer des **übergeordneten** objektes ,  also z.B bei rooms für project-1 die nummer 1 (ohne “project“)  Anmerkung: Bie „projects“ ist dieser wert als einziger sinnlos, kann aber als dummywert mit beliebigem inhalt mitgeliefert werden |
| Cx | Create generell | Detto für alle sonstigen “listtypes” mit den entsprechenden “specifications” | {“status”:0 , “message”:”Failed to update Item”}  {“status”:1 , “message”:”Item <item\_id> updated”}  Anmerkung: “itemid“ ist die Nummer des **übergeordneten** objektes ,  also z.B bei rooms für floor-10 die nummer 10 (ohne “floor“) , etc… |

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***SPECIFICATIONS*** | ***specification*** | Bemerkung |
|  | **Daten eines Objektes („listtype“: )** |  |  |
| S1 | projects | "specification": {  “name”: “Das Schöne Haus”,              "baumeister": "Harri Hurtig“,              "kapital": 1000  } | name Typ: String  baumeister Typ: String  kapital Typ: number |
| S2 | floors | "specification": {  “name”: “Das Schöne Stockwerk”,              "countFromBasement": 1  } | name Typ: String  countFromBasement Typ : number  Anm: als Ausnahme von der regel für die datenbank ist der feldname hier: floor\_count\_from\_basement |
| S3 | rooms | "specification": {    “name”: “Der Schöne Raum”,             "flaeche": 10  } | name Typ: String  flaeche [m2] Typ: number |
| S4 | devices | "specification": {  “name”: “Das Schöne Device”,  "fuseid“: 1,  „fname“:“DerNameDerSicherung“,  “devicetypeid”: 1,  „type“: „One of the predefined Types“  } | name Typ: String  fuseid Typ: number (die itemID der Sicherung)  devicetypeid Type: number (die id des Verbrauchertyps)  type: Typ: String (nur lesend, bei Bearbeiten wird nur die devicetypeid geschrieben) |
| S5 | sensors (Schalter) | "specification": {  “name”: “Der Schöne Sensor”,  "unit“: "A“: ,  "value“: 10,  “sensortypeid”: 1,  „type“: „One of the predefined Types“  } | name Typ: String  unit Typ: String [A, V, mA, … etc]  value Typ: number  sensortypeid Type: number (die id des Sensortyps)  type: Typ: String (nur lesend, bei Bearbeiten wird nur die sensortypeid geschrieben) |
| S6 | fis | "specification": {  “name”: “Der Schöne Fi”,  "current“: 500 ,  “fitypeid”: 1,  „type“: „One of the predefined Types“  } | name Typ: String  current Typ: numerisch [mA] Auslöse(fehler)strom  fitypeid Type: number (die id des FI-Typs)  type: Typ: String (nur lesend, bei Bearbeiten wird nur die fitypeid geschrieben) |
| S7 | fuses | "specification": {  “name”: “Die Schöne Sicherung”,  "current“: 16 ,  “fusetypeid”: 1,  „type“: „One of the predefined Types“  } | name Typ: String  current Typ: numerisch [A] Auslösestrom  fusetypeid Type: number (die id des Sicherungstyps)  type: Typ: String (nur lesend, bei Bearbeiten wird nur die fusetypeid geschrieben) |

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Kommunikationszweck*** | ***Request (POST)*** | ***Reply (Format: NEU)*** |
| **L** | **Anforderung von Shoppinglist und SicherungskreisListe -->** |  |  |
| L1 | Shoppinglist    Anmerkung: Enthält alle devices, sensors, fi, sicherungen; gleicher typ nur einmal mit anzahl (count) | { "data":  {"action":"shoppinglist”,”projectid”:1}  }  projectid 🡪 id des projektes für das die lieste erzeugt werden soll | “shoppinglist“: [{  "name": "Helligkeitssensor",  "count": "1"  }, {  "name": "Lichtschalter",  "count": "1"  }, {  "name": "Schalterchen",  "count": "2"  }, ……  ] |
| L2 | Sicherungskreisliste  Anmerkung: Enthält alle Stromkreise (fi, sicherungen) und die daran angeschlossenen Verbraucher (devices) | { "data":  {"action":"circuitlist” ,”projectid”:1}  }  projectid 🡪 id des projektes für das die lieste erzeugt werden soll | "circuits": [  {"name": "*FiXXX*",  "value": 100,  "fuses": [  {"name": "*SicherungXXX*",  "value": 16,  “fuseid”: 1,  "devices": [  "name": "*deviceXXX*",  "name": "*deviceXXX*", …… 🡪 weitere devices  ]  }, ……. 🡪 weitere Sicherungen  ]  }, ……. 🡪 weitere FI‘s  ] |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Kommunikationszweck*** | ***Request (POST)*** | ***Reply (Format: NEU)*** |
| **E** | **Fehlermeldungen** |  |  |
| E1 | Falscher wert in „action“ | { "data":  {"action":"somethingUnknown”}  } | {“status”:0 , “message”:”No such action”} |
| E2 | Falscher wert in „listtype“ | { "data":  {"action":"someValidListtype”,  "listtype":"somethingUnknown”}  } | {“status”:0 , “message”:”No such listtype”} |