JSON-Schema der statischen Produktdaten

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "properties": {
        "roomtypes": {
            "type": "array",
            "items": {
                "type": "object",
                "properties": {
                    "identifier": {
                        "type": "string"
                    },
                    "subMenuText": {
                        "type": "string"
                    },
                    "listText": {
                        "type": "string"
                    },
                    "pricingBasisText": {
                        "type": "string"
                    },
                    "capacity": {
                        "type": "integer"
                    }
                },
                "required": [
                    "identifier",
                    "subMenuText",
                    "listText",
                    "pricingBasisText",
                    "capacity"
                ]
            }
        },
        "boardings": {
            "type": "array",
            "items": {
                "type": "object",
                "properties": {
                    "identifier": {
                        "type": "string"
                    },
                    "listText": {
                        "type": "string"
                    },
                    "pricingBasisText": {
                        "type": "string"
                    }
                },
                "required": [
                    "identifier",
                    "listText",
                    "pricingBasisText"
                ]
            }
        },
        "specials": {
            "type": "array",
            "items": {
                "type": "object",
                "properties": {
                    "identifier": {
                        "type": "string"
                    },
                    "listText": {
                        "type": "string"
                    },
                    "pricingBasisText": {
                        "type": "string"
                    }
                },
                "required": [
                    "identifier",
                    "listText",
                    "pricingBasisText"
                ]
            }
        }
    },
    "required": [
        "roomtypes",
        "boardings",
        "specials"
    ]
}
```

JSON Schema der Preise und Verfügbarkeiten

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "array",
    "items": {
        "type": "object",
        "properties": {
            "identifier": {
                "type": "string"
            },
            "price": {
                "type": "string"
            }
        },
        "required": [
            "identifier",
            "price"
        ]
    }
}
```

JSON Schema des Warenkorbs

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "properties": {
        "checkInDate": {
            "type": "string"
        },
        "checkOutDate": {
            "type": "string"
        },
        "items": {
            "type": "array",
            "items": {
                "type": "object",
                "properties": {
                    "roomTypeIdentifier": {
                        "type": "string"
                    },
                    "roomTypeQuantity": {
                        "type": "integer"
                    },
                    "boardingIdentifier": {
                        "type": "string"
                    },
                    "specialsIdentifier": {
                        "type": "string"
                    }
                },
                "required": [
                    "roomTypeIdentifier",
                    "roomTypeQuantity",
                    "boardingIdentifier",
                    "specialsIdentifier"
                ]
            }
        }
    },
    "required": [
        "checkInDate",
        "checkOutDate",
        "items"
    ]
}
```

JSON Schema des Gesamtpreises

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "number"
}
```

JSON Schema des Warenkorbs mit Nutzerdaten

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "properties": {
        "checkInDate": {
            "type": "string"
        },
        "checkOutDate": {
            "type": "string"
        },
        "userFirstName": {
            "type": "string"
        },
        "userLastName": {
            "type": "string"
        },
        "userBirthDate": {
            "type": "string"
        },
        "userAddress": {
            "type": "string"
        },
        "userPlz": {
            "type": "string"
        },
        "userEmail": {
            "type": "string"
        },
        "alternateCheck": {
            "type": "boolean"
        },
        "userFirstNameAlternate": {
            "type": "string"
        },
        "userLastNameAlternate": {
            "type": "string"
        },
        "userBirthDateAlternate": {
            "type": "string"
        },
        "userAddressAlternate": {
            "type": "string"
        },
        "userPlzAlternate": {
            "type": "string"
        },
        "userEmailAlternate": {
            "type": "string"
        },
        "items": {
            "type": "array",
            "items": {
                "type": "object",
                "properties": {
                    "roomTypeIdentifier": {
                        "type": "string"
                    },
                    "roomTypeQuantity": {
                        "type": "integer"
                    },
                    "boardingIdentifier": {
                        "type": "string"
                    },
                    "specialsIdentifier": {
                        "type": "string"
                    }
                },
                "required": [
                    "roomTypeIdentifier",
                    "roomTypeQuantity",
                    "boardingIdentifier",
                    "specialsIdentifier"
                ]
            }
        }
    },
    "required": [
        "checkInDate",
        "checkOutDate",
        "userFirstName",
        "userLastName",
        "userBirthDate",
        "userAddress",
        "userPlz",
        "userEmail",
        "alternateCheck",
        "userFirstNameAlternate",
        "userLastNameAlternate",
        "userBirthDateAlternate",
        "userAddressAlternate",
        "userPlzAlternate",
        "userEmailAlternate",
        "items"
    ]
}
```

InitialDataController.php

```php
<?php

namespace AppBundle\Controller;

use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;
use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use AppBundle\Entity\Product;
use AppBundle\Entity\RoomType;
use AppBundle\Entity\AdditionalProduct;
use AppBundle\Entity\AdditionalProductCategory;

class InitialDataController extends Controller {

    /**
     * @Route("/api/v1/initialdata", name="initialdata_v1")
     */
```

```php
public function initialDataAction_v1(Request $request) {

    $em = $this->getDoctrine()->getManager();

    $query = $em->createQuery('
            SELECT

            p.identifier ,
            p.subMenuText ,
            p.listText ,
            p.pricingBasisText ,
            p.capacity

            FROM AppBundle:RoomType p

            ORDER BY p.positionInSubMenu ASC

            ');

    $roomtypes = $query->getResult();

    $query = $em->createQuery("
            SELECT

            a.identifier ,

            a.listText ,
            a.pricingBasisText

            FROM AppBundle:AdditionalProduct a
            JOIN a.additionalproductcategory p
            WHERE p.identifier = 'boardings'

            ORDER BY a.positionInList ASC

            ");

    $boardings = $query->getResult();

    $query = $em->createQuery("
            SELECT

            a.identifier ,

            a.listText ,
            a.pricingBasisText

            FROM AppBundle:AdditionalProduct a
            JOIN a.additionalproductcategory p
            WHERE p.identifier = 'specials'

            ORDER BY a.positionInList ASC

            ");

    $specials = $query->getResult();


    $products = array(
        'roomtypes' => $roomtypes,
        'boardings' => $boardings,
        'specials' => $specials
    );

    $productsJSON = json_encode($products, 320); // 320 : 0000000101000000 = 256 + 64 :
    JSON_UNESCAPED_SLASHES => 64 + JSON_UNESCAPED_UNICODE => 256

    $resp = new Response($productsJSON);
    $resp->headers->set('Content-Type', 'application/json ; charset=utf-8');

    return $resp;
}
```

```
}
```

TotalPriceController.php

```php
<?php

namespace AppBundle\Controller;

use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;
use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use AppBundle\Entity\Cart;
use AppBundle\Entity\Item;
use AppBundle;

class TotalPriceController extends Controller {

    /**
     * @Route("/api/v1/totalPrice", name="totalPrice_v1")
     */
    public function totalPriceAction_v1(Request $request) {

        $input = $request->getContent();

        $em = $this->getDoctrine()->getManager();

        if (
                (!$input)
        ) {


            $resp = new Response(
                    "Malformed request syntax. "
                    . " "
            );
            $resp->setStatusCode(Response::HTTP_BAD_REQUEST);
            $resp->headers->set('Content-Type', 'Content-Type: text/html; charset=utf-8');
            return $resp;
        }

        $input_sanitized = str_replace(array("\n", "\t", "\r"), '', $input); // remove newlines , tabs , carriage
                                                                              return

        $json_decoded = json_decode($input_sanitized, false); // false -> object , true -> array

        $c = new Cart();
        $c->setCheckInDate($json_decoded->checkInDate);
        $c->setCheckOutDate($json_decoded->checkOutDate);
        foreach ($json_decoded->items as $item) {
            $i = new Item();
            $i->setRoomTypeIdentifier($item->roomTypeIdentifier);
            $i->setRoomTypeQuantity($item->roomTypeQuantity);
            $i->setBoardingIdentifier($item->boardingIdentifier);
            $i->setSpecialIdentifier($item->specialsIdentifier);
            $c->addItem($i);
        }

        $totalPrice = $em->getRepository('AppBundle:Cart')->calculateTotalPrice($c); // Zugriff über
                                                                                      EntityRepository

        $totalPriceJSON = json_encode($totalPrice, 320); // 320 : 0000000101000000 = 256 + 64 :
        JSON_UNESCAPED_SLASHES => 64 + JSON_UNESCAPED_UNICODE => 256

        $resp = new Response($totalPriceJSON);
        $resp->headers->set('Content-Type', 'application/json ; charset=utf-8');

        return $resp;
    }
}
```

Product.php

```
<?php

namespace AppBundle\Entity;

use Doctrine\ORM\Mapping as ORM;
use Doctrine\Common\Collections\ArrayCollection;

/**
 * @ORM\Entity(repositoryClass="AppBundle\Entity\ProductRepository")
 * @ORM\Table(name="product")
 * @ORM\InheritanceType("JOINED")
 * @ORM\DiscriminatorColumn(name="discriminator", type="string")
 * @ORM\DiscriminatorMap({"product" = "Product", "roomType" = "RoomType" , "additionalProduct" =
"AdditionalProduct"})
 */
abstract class Product {

    /**
     * @ORM\Column(type="integer")
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="AUTO")
     */
    private $id;

    /**
     * @ORM\Column(type="string" , length=100 , nullable=false , unique=true)
     */
    private $identifier;

    /**
     * @ORM\Column(type="string" , length=100 , nullable=true)
     */
    private $listText; // fuer list

    /**
     * @ORM\Column(type="string" , length=100 , nullable=true)
     */
    private $pricingBasis;

    /**
     * @ORM\Column(type="string" , length=100 , nullable=true)
     */
    private $pricingBasisText;

    /**
     * @ORM\OneToMany(targetEntity="Price", mappedBy="product")
     */
    private $prices; // ArrayCollection

    public function __construct() {
        $this->prices = new ArrayCollection();
    }

    /**
     * @ORM\ManyToOne(targetEntity="Hotel", inversedBy="products")
     * @ORM\JoinColumn(name="hotel_id", referencedColumnName="id")
     */
    private $hotel;

    /**
     * Get id
     *
     * @return integer
     */
    public function getId() {
        return $this->id;
    }
```

```
/**
 * Set identifier
 *
 * @param string $identifier
 *
 * @return Product
 */
public function setIdentifier($identifier) {
    $this->identifier = $identifier;

    return $this;
}

/**
 * Get identifier
 *
 * @return string
 */
public function getIdentifier() {
    return $this->identifier;
}

/**
 * Set listText
 *
 * @param string $listText
 *
 * @return Product
 */
public function setListText($listText) {
    $this->listText = $listText;

    return $this;
}

/**
 * Get listText
 *
 * @return string
 */
public function getListText() {
    return $this->listText;
}

/**
 * Set pricingBasis
 *
 * @param string $pricingBasis
 *
 * @return Product
 */
public function setPricingBasis($pricingBasis) {
    $this->pricingBasis = $pricingBasis;

    return $this;
}

/**
 * Get pricingBasis
 *
 * @return string
 */
public function getPricingBasis() {
    return $this->pricingBasis;
}

/**
 * Set pricingBasisText
 *
 * @param string $pricingBasisText
 *
```

```
     * @return Product
     */
    public function setPricingBasisText($pricingBasisText) {
        $this->pricingBasisText = $pricingBasisText;

        return $this;
    }

    /**
     * Get pricingBasisText
     *
     * @return string
     */
    public function getPricingBasisText() {
        return $this->pricingBasisText;
    }

    /**
     * Add price
     *
     * @param \AppBundle\Entity\Price $price
     *
     * @return Product
     */
    public function addPrice(\AppBundle\Entity\Price $price) {
        $this->prices[] = $price;

        return $this;
    }

    /**
     * Remove price
     *
     * @param \AppBundle\Entity\Price $price
     */
    public function removePrice(\AppBundle\Entity\Price $price) {
        $this->prices->removeElement($price);
    }

    /**
     * Get prices
     *
     * @return \Doctrine\Common\Collections\Collection
     */
    public function getPrices() {
        return $this->prices;
    }

    /**
     * Set hotel
     *
     * @param \AppBundle\Entity\Hotel $hotel
     *
     * @return Product
     */
    public function setHotel(\AppBundle\Entity\Hotel $hotel = null) {
        $this->hotel = $hotel;

        return $this;
    }

    /**
     * Get hotel
     *
     * @return \AppBundle\Entity\Hotel
     */
    public function getHotel() {
        return $this->hotel;
    }

}
```

RoomType.php

```php
<?php

namespace AppBundle\Entity;

use Doctrine\ORM\Mapping as ORM;
use Doctrine\Common\Collections\ArrayCollection;
use JsonSerializable;

/**
 * @ORM\Entity
 * @ORM\Table(name="roomType")
 */
class RoomType extends Product {

    /**
     * @ORM\Column(type="integer")
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="AUTO")
     */
    private $id;

    /**
     * @ORM\Column(type="smallint", nullable=true)
     */
    private $capacity;

    /**
     * @ORM\Column(type="smallint" , nullable=true)
     */
    private $quantityOfPersons;

    /**
     * @ORM\Column(type="smallint" , nullable=true)
     */
    private $positionInSubMenu;

    /**
     * @ORM\Column(type="string" , length=100 , nullable=true)
     */
    private $subMenuText; // fuer submenu

    /**
     * @ORM\OneToMany(targetEntity="Availability", mappedBy="roomType")
     */
    private $availabilities; // ArrayCollection

    public function __construct() {
        parent::__construct(); // Product Konstruktor aufrufen
        $this->availabilities = new ArrayCollection();
    }

    /**
     * Get id
     *
     * @return integer
     */
    public function getId() {
        return $this->id;
    }

    /**
     * Set capacity
     *
     * @param integer $capacity
     *
     * @return RoomType
     */
    public function setCapacity($capacity) {
        $this->capacity = $capacity;
    return $this;
}

/**
 * Get capacity
 *
 * @return integer
 */
public function getCapacity() {
    return $this->capacity;
}

/**
 * Set quantityOfPersons
 *
 * @param integer $quantityOfPersons
 *
 * @return RoomType
 */
public function setQuantityOfPersons($quantityOfPersons) {
    $this->quantityOfPersons = $quantityOfPersons;

    return $this;
}

/**
 * Get quantityOfPersons
 *
 * @return integer
 */
public function getQuantityOfPersons() {
    return $this->quantityOfPersons;
}

/**
 * Set positionInSubMenu
 *
 * @param integer $positionInSubMenu
 *
 * @return RoomType
 */
public function setPositionInSubMenu($positionInSubMenu) {
    $this->positionInSubMenu = $positionInSubMenu;

    return $this;
}

/**
 * Get positionInSubMenu
 *
 * @return integer
 */
public function getPositionInSubMenu() {
    return $this->positionInSubMenu;
}

/**
 * Set subMenuText
 *
 * @param string $subMenuText
 *
 * @return RoomType
 */
public function setSubMenuText($subMenuText) {
    $this->subMenuText = $subMenuText;

    return $this;
}

/**
 * Get subMenuText
 *
```

```
     * @return string
     */
    public function getSubMenuText() {
        return $this->subMenuText;
    }

    /**
     * Add availability
     *
     * @param \AppBundle\Entity\Availability $availability
     *
     * @return RoomType
     */
    public function addAvailability(\AppBundle\Entity\Availability $availability) {
        $this->availabilities[] = $availability;

        return $this;
    }

    /**
     * Remove availability
     *
     * @param \AppBundle\Entity\Availability $availability
     */
    public function removeAvailability(\AppBundle\Entity\Availability $availability) {
        $this->availabilities->removeElement($availability);
    }

    /**
     * Get availabilities
     *
     * @return \Doctrine\Common\Collections\Collection
     */
    public function getAvailabilities() {
        return $this->availabilities;
    }

}
```

LoadPriceData.php

```
<?php

namespace AppBundle\DataFixtures\ORM;

use Doctrine\Common\DataFixtures\AbstractFixture;
use Doctrine\Common\DataFixtures\OrderedFixtureInterface;
use Doctrine\Common\Persistence\ObjectManager;
use AppBundle\Entity\Price;

class LoadPriceData extends AbstractFixture implements OrderedFixtureInterface {

    public function load(ObjectManager $manager) {

        $pr1 = new Price();
        $pr1->setDate(new \DateTime("2016-01-01"));
        $pr1->setValue(110.00);
        $pr1->setProduct($this->getReference('singleroom'));
        $manager->persist($pr1);
        $manager->flush();
        unset($pr1);

        $pr1a = new Price();
        $pr1a->setDate(new \DateTime("2016-01-15"));
        $pr1a->setValue(120.00);
        $pr1a->setProduct($this->getReference('singleroom'));
        $manager->persist($pr1a);
        $manager->flush();
        unset($pr1a);

        $pr1b = new Price();
```

```php
$pr1b->setDate(new \DateTime("2016-02-01"));
$pr1b->setValue(130.00);
$pr1b->setProduct($this->getReference('singleroom'));
$manager->persist($pr1b);
$manager->flush();
unset($pr1b);

$pr2 = new Price();
$pr2->setDate(new \DateTime("2016-01-01"));
$pr2->setValue(120.00);
$pr2->setProduct($this->getReference('doubleroom'));
$manager->persist($pr2);
$manager->flush();
unset($pr2);

$pr3 = new Price();
$pr3->setDate(new \DateTime("2016-01-01"));
$pr3->setValue(130.00);
$pr3->setProduct($this->getReference('twinroom'));
$manager->persist($pr3);
$manager->flush();
unset($pr3);

$pr4 = new Price();
$pr4->setDate(new \DateTime("2016-01-01"));
$pr4->setValue(140.00);
$pr4->setProduct($this->getReference('tripleroom'));
$manager->persist($pr4);
$manager->flush();
unset($pr4);

$pr5 = new Price();
$pr5->setDate(new \DateTime("2016-01-01"));
$pr5->setValue(150.00);
$pr5->setProduct($this->getReference('familyroom'));
$manager->persist($pr5);
$manager->flush();
unset($pr5);

$pr6 = new Price();
$pr6->setDate(new \DateTime("2016-01-01"));
$pr6->setValue(160.00);
$pr6->setProduct($this->getReference('apartmentsingle'));
$manager->persist($pr6);
$manager->flush();
unset($pr6);

$pr7 = new Price();
$pr7->setDate(new \DateTime("2016-01-01"));
$pr7->setValue(170.00);
$pr7->setProduct($this->getReference('apartmentdouble'));
$manager->persist($pr7);
$manager->flush();
unset($pr7);

$pr8 = new Price();
$pr8->setDate(new \DateTime("2016-01-01"));
$pr8->setValue(12.50);
$pr8->setProduct($this->getReference('halfpension'));
$manager->persist($pr8);
$manager->flush();
unset($pr8);

$pr9 = new Price();
$pr9->setDate(new \DateTime("2016-01-01"));
$pr9->setValue(25.00);
$pr9->setProduct($this->getReference('fullpension'));
$manager->persist($pr9);
$manager->flush();
unset($pr9);

$pr10 = new Price();
```

```
        $pr10->setDate(new \DateTime("2016-01-01"));
        $pr10->setValue(8.00);
        $pr10->setProduct($this->getReference('breakfast'));
        $manager->persist($pr10);
        $manager->flush();
        unset($pr10);

        $pr12 = new Price();
        $pr12->setDate(new \DateTime("2016-01-01"));
        $pr12->setValue(0.00);
        $pr12->setProduct($this->getReference('noboarding'));
        $manager->persist($pr12);
        $manager->flush();
        unset($pr12);

        $pr13 = new Price();
        $pr13->setDate(new \DateTime("2016-01-01"));
        $pr13->setValue(12.50);
        $pr13->setProduct($this->getReference('champagnebreakfast'));
        $manager->persist($pr13);
        $manager->flush();
        unset($pr13);

        $pr14 = new Price();
        $pr14->setDate(new \DateTime("2016-01-01"));
        $pr14->setValue(30.00);
        $pr14->setProduct($this->getReference('rosesinrooms'));
        $manager->persist($pr14);
        $manager->flush();
        unset($pr14);

        $pr15 = new Price();
        $pr15->setDate(new \DateTime("2016-01-01"));
        $pr15->setValue(25.00);
        $pr15->setProduct($this->getReference('raftingtour'));
        $manager->persist($pr15);
        $manager->flush();
        unset($pr15);
    }

    public function getOrder() {
        // the order in which fixtures will be loaded
        // the lower the number, the sooner that this fixture is loaded
        return 7;
    }

}
```

PriceRepository.php

```
<?php

namespace AppBundle\Entity;

use DateTime;
use DateInterval;
use DatePeriod;

/**
 * PriceRepository
 *
 * This class was generated by the Doctrine ORM. Add your own custom
 * repository methods below.
 */
class PriceRepository extends \Doctrine\ORM\EntityRepository { // returns price object

    public function findLatestPricePerProductAndDateTime(Product $prod, DateTime $date) {

        $em = $this->getEntityManager();

        $query = $em->createQuery(
            'SELECT pri FROM AppBundle:price pri JOIN pri.product prod WHERE prod.identifier = ?1 ORDER BY
                    pri.date DESC'
    );
    $query->setParameter(1, $prod->getIdentifier());

    $result = $query->getResult(); // Array of Price Objects

    foreach ($result as $r) { // laufe durch ResultSet vom Neuesten zum Aeltesten
        if ($r->getDate() <= $date) {
            return $r;
        }
    }

    return $result[0];
}

public function calculateTotalAmountPerProductAndDateInterval(Product $prod, DateTime $checkIn, DateTime
                                                                    $checkOut) { // returns double
    $em = $this->getEntityManager();

    $sum = 0;

    $interval = new DateInterval('P1D'); // 1 Tag
    $daterange = new DatePeriod($checkIn, $interval, $checkOut);

    foreach ($daterange as $date) {
        $sum += $em->getRepository('AppBundle:Price')->findLatestPricePerProductAndDateTime($prod, $date)-
                                                                                            >getValue();
    }

    return $sum;
}

public function calculatePriceAveragePerProductAndDateInterval(Product $prod, DateTime $checkIn, DateTime
                                                                            $checkOut) { // returns double
    $em = $this->getEntityManager();

    $sum = 0;
    $days = 0;
    $avg = 0;

    $interval = new DateInterval('P1D'); // 1 Tag
    $daterange = new DatePeriod($checkIn, $interval, $checkOut);

    foreach ($daterange as $date) {
        $days += 1;
        $sum += $em->getRepository('AppBundle:Price')->findLatestPricePerProductAndDateTime($prod, $date)-
                                                                                            >getValue();
    }

    $avg = round($sum / $days, 2); // auf 2 Nachkommastellen gerundet

    return $avg; // arithmetischer Mittelwert
}

public function findLatestPricePerProductIdentifierAndDateTime($productString, DateTime $date) { // returns
                                                                                            price object
    $em = $this->getEntityManager();

    $query = $em->createQuery(
            'SELECT prod FROM AppBundle:product prod WHERE prod.identifier = ?1'
    );
    $query->setParameter(1, $productString);

    $productObject = $query->getResult()[0]; // 1 ProductObject

    $query = $em->createQuery(
            'SELECT pri FROM AppBundle:price pri JOIN pri.product prod WHERE prod.identifier = ?1 ORDER BY
                    pri.date DESC'
    );
    $query->setParameter(1, $productObject->getIdentifier());
```

```
        $result = $query->getResult(); // Array of Price Objects

        foreach ($result as $r) { // laufe durch ResultSet vom Neuesten zum Aeltesten
            if ($r->getDate() <= $date) {
                return $r;
            }
        }

        return $result[0];
    }

}
```