TYPO3 Core - Bug #68651

Epic # 77562 (Accepted): Misbehaviors with datetime values and timezones

Datetime() properties have wrong timezone

2015-07-30 21:02 - Martin Bless

Status:	Accepted	**Start date:**	2015-07-30
Priority:	Must have	**Due date:**	
Assignee:	Andreas Wolf	**% Done:**	0%
Category:	Extbase	**Estimated time:**	0.00 hour
Target version:	Candidate for patchlevel		
TYPO3 Version:	7	**Complexity:**	
PHP Version:		**Is Regression:**	No
Tags:		**Sprint Focus:**	

Description

Summary:

The TYPO3 backend does it right. Would be great to have Extbase doing right as well!

Details:

My timezone is set to "Europe/Berlin" ('phpTimeZone' => 'Europe/Berlin')

In the backend everything is ok: I can enter "31-10-2015 00:00" in a field 'firstDay' and it's always correct. But in the frontend, with Extbase/Fluid and

```
<f:format.date format="d.m.Y">{offer.firstDay}</f:format.date>`
```

I will see "30.10.2015" as output, because that date has been calculated for timezone GMT/UTC, which is "30-10-2015 22:00" which is 2 hours less than our current CEST (central european summer time).

The problem is, that Extbase instantiates the offer.firstDay DateTime() object with timezone_type => 1:

```
object(DateTime)[1]
  public 'date' => string '2015-10-30 22:00:00' (length=19)
  public 'timezone_type' => int 1
  public 'timezone' => string '+00:00' (length=6)
```

while it actually needs the correct timezone information.

I can correct that in the following manner:

```
// we use $now to copy the timezone from
$now = new \DateTime();
firstDay->setTimeZone($now->getTimezone());
```

After that procedure var_dump($firstDay) says:

```
object(DateTime)[1]
  public 'date' => string '2015-10-31 00:00:00' (length=19)
  public 'timezone_type' => int 3
  public 'timezone' => string 'Europe/Berlin' (length=13)
```

And now $firstDay will give the same result that the backend gives for

```
<f:format.date format="d.m.Y">{offer.firstDay}</f:format.date>`
```

The result is now the expected "31.10.2015".

Earlier discussions

When searching for a solution I found an earlier discussion of that issue here:
http://lists.typo3.org/pipermail/typo3-project-typo3v4mvc/2010-July/006013.html

Related issues:

Related to TYPO3 Core - Bug #72878: wrong datetime handling, they are not UTC...	**Closed**	**2016-01-21**
Related to TYPO3 Core - Bug #80008: Avoid warning dialog when closing records...	**Closed**	**2017-02-24**
Related to TYPO3 Core - Feature #80559: datetime and time handling should alw...	**Closed**	**2017-03-29**
Related to TYPO3 Core - Bug #80679: Input field eval datetime in FlexForm con...	**Closed**	**2017-04-04**
Related to TYPO3 Core - Bug #80349: Extbase mapping of \DateTime for integer ...	**New**	**2017-03-20**
Related to TYPO3 Core - Bug #68849: Unstable persistence handling of DateTime...	**Closed**	**2015-08-07**
Related to TYPO3 Core - Bug #79613: Saving wrong Date into DB if Field is of ...	**New**	**2017-02-03**

History

#1 - 2015-08-05 13:59 - Peter Gisler

May I add the following thoughts...

I encountered the same problem. In my domain object I have a property of type \DateTime:

```
/**
 * @var \DateTime
 */
protected $sendDate = NULL;
```

Upon creating a new object I add a DateTime object:

```
$moment = new \DateTime('2015-08-05 10:00:00'); // timezone of server: Europe/Zurich
$object->setSendDate($moment);
```

After persisting this object I can read in my database (column-type: datetime) the exact same value: 2015-08-05 10:00:00

When I retrieve this object back from the database and var_dump the sendDate property, I get a \DateTime object with timezone 'Europe/Zurich'. But now the value of this objects sendDate-property is 2015-08-05 12:00:00 (!)

I'm therefore assuming that extbase expects a datetime database value to be in UTC and upon retrieving such a value is converting it to the "correct" timezone. The problem is now, that during the persisting process, that value never gets transformed to UTC!

#2 - 2015-10-20 15:48 - Teamgeist Medien

Can confirm this problem in TYPO3 6.2.15... dangerous shit...

#3 - 2016-05-25 10:21 - Christoph Dörfel

Just stumbled onto this bug too. Timezone is not respected when saving values to the database.
bump

#4 - 2016-07-25 11:07 - Stefan Froemken

I can't confirm this bug directly, but I can confirm that TYPO3 could have problems with DST. DST handling will work correctly, if Timezone-Type is of type 3 only.

```
$date1 = new \DateTime(date('c', $value)); // TYPO3 way: TZ type 1 with +02:00
$date2 = new \DateTime('@' . $value); // previous version in TYPO3: Plain UTC. Difference is always +00:00. So
 time will be set -2 hours for Germany
$date3 = new \DateTime(date('r', $value)); // Same result as date('c')
$date5 = new \DateTime(date('Y-m-d H:i:s', $value)); // TZ type 3. No TZ given, so default TZ (Europe/Berlin)
will be set
$date6 = new \DateTime(date('Y-m-d H:i:s e', $value)); // e represents TZ Europe/Berlin. TZ type 3.
$date7 = new \DateTime(date('Y-m-d H:i:s P', $value)); // P represents TZ difference like +02:00. TZ type 1
$date8 = new \DateTime(date('Y-m-d T H:i:s', $value)); // TZ represents TZs like EST, MDT, CEST, CET. TZ type
2
```

#5 - 2016-07-25 11:20 - Martin Bless

To be more specific:

Version:
At the time of the bug report, July 2015, it was TYPO3 version 6.2.15

Domain model:

```
/**
 * firstDay
 *
 * @var \DateTime
 * @validate NotEmpty
 */
protected $firstDay = NULL;
```

TCA:

```
'first_day' => array(
   'exclude' => 0,
   'label' => 'LLL:EXT:bwoffers/Resources/Private/Language/locallang_db.xlf:tx_bwoffers_domain_model_offer.fir
st_day',
   'config' => array(
      'type' => 'input',
      'size' => 10,
      'eval' => 'date',
      'checkbox' => 1,
      'default' => strtotime('next Monday'),
      'placeholder' => 'dd-mm-yyyy'
   ),
),
```

#6 - 2016-08-21 21:03 - Oliver Hader

- Parent task set to #77562

#7 - 2016-09-02 11:30 - Andreas Wolf

- Status changed from New to Resolved

This was most likely fixed with 827219a1c35b4aca6dbab5855a36e9277b2ec8f4 (on master) and on 7.6 already, too.

#8 - 2016-09-07 09:34 - Jacco van der Post

This annoying bug is not fixed in TYPO3 7.6.10

Fluid outputs time one hour later than backend.

#9 - 2016-09-12 15:37 - Peter Gisler

I'm using version 7.6.10 and can confirm: This bug is NOT fixed yet! :-(

#10 - 2016-09-15 12:38 - Heinrich Pegelow

- TYPO3 Version changed from 7 to 6.2

The bug is still in Typo3 6.2.27.

Untill this is fixed, I changed all my GetFunktions like this:

```
/**
 * Returns the eintrittgewaehrt
 *
 * @return \DateTime $eintrittgewaehrt
     */
    public function getEintrittgewaehrt() {
        if ($this->eintrittgewaehrt != NULL) {
            $this->eintrittgewaehrt->modify('-2 hours');
        }
        return $this->eintrittgewaehrt;
    }
```

And of cource I have to modify the "-2 hours" when DayLightSavingTime changes in end of october.

#11 - 2016-09-15 12:45 - Christoph Bernhard

Please change the status of this issue! It is not nearly resolved! What a pain - unbelievable...

#12 - 2016-09-19 14:01 - Bernhard Berger

- TYPO3 Version changed from 6.2 to 7

Christoph Bernhard wrote:

> Please change the status of this issue! It is not nearly resolved! What a pain - unbelievable...

Agreed. This isn't resolved in any way. Still persists in the latest 7 releases too... Seems like they just set some issues to `resolved` these days that they just don't want to fix.. this isn't the first one I encountered lately..

#13 - 2016-11-11 16:21 - Mathias Schreiber

- Status changed from Resolved to New

- Assignee set to Andreas Wolf

Hey Andy,

can to take another look?

@Everybody ranting in this issue:
Here's how redmine works:
It does NOT send emails to anybody on closed or resolved tickets.
So unfortunately the workflow has to be like this:
Get in touch with the person that closed the ticket (Email, Slack, Twitter, Bushdrums, Telephone or alike) and state that the issue is not resolved as expected.

Ranting in a closed ticket gets you nowhere :)

#14 - 2016-11-21 12:41 - Arno Schoon

Hi Andreas,

we ran into a similar issue so I did some testing, it seems a date value is persisted (in \TYPO3\CMS\Core\DataHandling\DataHandler::process_datamap) after a conversion using `gmdate`. The value stored should be a unix time based on UTC, but when `\TYPO3\CMS\Extbase\Persistence\Generic\Mapper\DataMapper::mapDateTime` is used to convert the value the current timezone (in my case Europe/Amsterdam) seems to fail. Forcing the timezone to UTC fixes the issue when formatting the date.

The problem seems to occur if the TCA eval is `time`, not sure if all checks in the conversion process match. What also made me wonder was the comment "integer timestamps are local server time" in `mapDateTime()`, no further conversion takes place here.

Rgs,
Arno

#15 - 2016-12-09 14:41 - Jörg Velletti

- File DateViewHelper.php added

The Bug is in my Opinion in the Fluid Viewhelper
http/typo3/sysext/fluid/Classes/ViewHelpers/Format/DateViewHelper.php

it does not read The Globals TYPO3_CONF_VARS and takes by default date_default_timezone_get() and does no Fallback to UTC

It could be solved with 1 line of code and one change.

```
$timeZone = trim($GLOBALS['TYPO3_CONF_VARS']['SYS']['phpTimeZone']) == '' ? date_default_timezone_get()  : $ti
meZone ;
```

and replace date_default_timezone_get() with $timezone

```
  $date->setTimezone(new \DateTimeZone($timeZone));
```

But also this Viewhelper LACKS support for a USER spezific TimeZone setting:

f.e. a visitor in south America is watching the Event list of Online Webinar Session from our dependancy in Madrid Spain.
he will actually see what Time ?
timeZone where it happens (the correct one for my colueges in Madrid)?
The timeZone of the Server (it is in Stassburg)

or isnt it better to show it in his TimeZone? so an additional argument timeZone should be added to the dateViewhelper

See attached my version of this DateViewhelper

it has everything that Install Tool decribes: allow setting in Conf Vars, fallback to PPHP Settings or, if still does not work use UTC with an additional overwriting in a specific Viewhelper, user/Browser depening! I just add the new argument (default = null) and instead of one line setting of $timeZone i have 7 lines.

```
    $timeZone = $arguments['timeZone'] === null ? trim($GLOBALS['TYPO3_CONF_VARS']['SYS']['phpTimeZone'])  : $
arguments['timeZone'];
    if( ! in_array($timeZone, timezone_identifiers_list()) ) {
        $timeZone = date_default_timezone_get() ;
    }
    if( ! in_array($timeZone, timezone_identifiers_list()) ) {
        $timeZone = 'UTC' ;
    }
```

maybe some one who has already gerrit running can copy my code, compare it with actual repository and do the next steps ..

#16 - 2017-01-31 08:12 - Einar Gislason

I'm not sure, but I suspect this issue has had the unfortunate consequence of making our pages fail the JS validation each time they're saved.
Almost all our pages have starttime set. When we edit some field in the BE form (title f.ex.) and save, the starttime field will actually get the class of has-change, even though the value in the field doesn't appear to change at all.
Comparing the value in the BE form (and thus in $fieldArray) to the value in the DB shows a difference of one hour in the value.

To reproduce:
- Create a page with a title and starttime value (save and close)
- Open the page again, edit the title and save (just save, not save and close)
- Now check the starttime field - it will have a class of has-change.
- If you try and close the form now you will get a warning telling you there are unsaved changes. This is because the values in the form and DB don't match, I suspect.

#17 - 2017-02-25 12:15 - Joerg Kummer

See also #80008

#18 - 2017-04-15 00:18 - Mona Muzaffar

- Related to Bug #80679: Input field eval datetime in FlexForm converts to wrong timestamp added

#19 - 2017-04-15 00:34 - Mona Muzaffar

- Related to Epic #80852: Datetime handling in backend added

#20 - 2017-05-29 14:38 - Paul Golmann

- Target version set to next-patchlevel

I've created a patch that changes the type of timezone from offset to region name for dateTimes read from timestamp database fields.
Please have a look at #80349 and https://review.typo3.org/52952.

Greetings. Paul

#21 - 2017-05-29 14:49 - Paul Golmann

- Related to Bug #80349: Extbase mapping of \DateTime for integer values does not set timezone with region string but only offset added

#22 - 2017-05-29 14:49 - Paul Golmann

- Related to Bug #80349: Extbase mapping of \DateTime for integer values does not set timezone with region string but only offset added

#23 - 2017-05-29 14:49 - Paul Golmann

- Related to deleted (Bug #80349: Extbase mapping of \DateTime for integer values does not set timezone with region string but only offset)

#24 - 2017-06-06 19:12 - Markus Klein

- Status changed from New to Accepted

#25 - 2017-06-06 19:12 - Markus Klein

- Related to Bug #68849: Unstable persistence handling of DateTime (don't get what you set) added

#26 - 2017-06-06 19:24 - Markus Klein

- Related to Bug #79613: Saving wrong Date into DB if Field is of type DATE added

#27 - 2017-06-07 13:01 - Riccardo De Contardi

- Related to deleted (Epic #80852: Datetime handling in backend)

#28 - 2017-08-11 08:27 - SICOR KDL GmbH

Hello,

I guess this (8.7.4) observation is related:

I have a TCA with an input, eval = time.

In the backend I save 09:00.
In the database it writes 32400 (9*3600 => correct!)
In the frontend with f:format.date I get 10:00. Eh...

This seems to be no ~~rocket~~ timezone science, but a matter of simple math: Divide the value by 3600 for hours and the rest by 60 for minutes.

I assume the error occurs because it internally mistakenly uses some date object, which naturally assumes the 01.01.1970 for its math, which is a day in winter(!) and so currently one hour off in summer...

Can't f:format.date branch out differently, if the value is less than 86400? Or maybe provide a viewhelper f:format.time?

Greetings,
Manuel

#29 - 2017-08-11 09:37 - Jörg Velletti

I have investigated this issue yesterday a little bit deaper:

It is a little bit more than just a math Problem:

first influence:
php.ini -> What is the ORIGINAL @date_default_timezone_get()
test this in a separate php file not INSIDE of a TYPO3 Script as TYPO3 is changing this!

second influence:
TYPO3 Localconfiguration.php -> is $GLOBALS['TYPO3_CONF_VARS']['SYS']['phpTimeZone'] set?

third influence
during the TYPO3 Bootstrap Process @date_default_timezone_set() is Called.
it will be set either with $GLOBALS['TYPO3_CONF_VARS']['SYS']['phpTimeZone'], Value from php.ini or is set as Fallback to 'UTC'

last influence:
Extbase and Fluid Data mapper and the TYPO3 Core:
DateTime Values in the ExtBase Datamapper are in many Cases using as HARDCODED Fallback the UTC timezone , in Core some places uses $GLOBALS['TYPO3_CONF_VARS']['SYS']['phpTimeZone'] and on other places the value from @date_default_timezone_get()

Solution can be:
REMOVE all hard Coded 'UTC' calls and rely only @date_default_timezone_set() in EXTbase mapDateTime
or
Add a DateTime Parameter to the Fluid DateTime ViewHelper (best Approach in my opinion as with this, everyone can help himself in the end)

I Actually have
php.ini with "Europe/Berlin"

I Added this
$GLOBALS['TYPO3_CONF_VARS']['SYS']['phpTimeZone'] = "UTC"

ALL Entered Values in Frontend are now Wrong! -> I Fixed the wrong Date Fields with some SQL Statements so OLD Values in Frontend look correct

Entering a DateTime Value in the backend is now also shown Correctly in Frontend.
Also Summertime/Wintertime Values Are Working correctly.

The only problem with THIS Approach is comparasions between actual server time and database value ..
(f.e.
// has this event startet
$now = new \dateTime(), will deliver a UTC DateTime Value means actuall 2 hours too early.

So : if you have problems with this , check also your php settings and the TYPO value. Maybe setting ['phpTimeZone'] = "UTC" can help. but please check you already entered date Time values (startdate, enddate of tt_content elements , pages, Event databes entry, scheduler tasks .. and more ..)

#30 - 2017-08-11 10:01 - Martin Bless

Hello Jörg (Velletti):

I have investigated this issue yesterday a little bit deaper:

Thanks a lot! That sounds like you did a major step in clarifying this issue!

But of course: If the logic in the core isn't uniform that needs correction!

#31 - 2017-08-11 10:38 - Jörg Velletti

maybe .. but finally changing the datamapper Code can brake dateValues on some users TYPo3 db if PHP.ini and TYPo3 setting ist diffent from UTC ..
For me only adding a timeZone="" parameter to the Fluid format.date() Viewhelper can be the best for all solution.

Why? I have a Server in Strassburg and reside in Munich. so the serverstimezone is Europe/Berlin.
but we have a spanish dependency, organizing online trainings for the spanish speaking Customers in Madrid. With Cusotmers in Gran canaria, Mexico ... .

They all get dateTime Values in Frontent for BERLIN .. (and have to calculate on their own)

#32 - 2017-12-12 13:09 - Stefan P

We encountered this problem, too. But Fluid/Extbase seem to work correctly here. The problem is that the Backend has no possibility to configure/edit the timezone of time inputs.

Timestamps are seconds from 1970 when located on the prime meridian (UTC), but when *displaying* them on any other meridian steps of 3600 seconds have to be added/substracted (but only for *display*). But for eval = time fields no timezone considerations are done in the backend at all. So "9:00" is stored as 9*3600 regardless of where the backend user is located (i.e. what his browser sends for the region) or what the configured server timezone is. For eval = date or eval = datetime this timezone considerations seem to happen, however.

For PHP \DateTime objects there is no distinction between "date" and "time" and "datetime". A \DateTime object is only reperesented by an UTC timestamp + a timezone (which is correct: it has the actual time and information for a potential display offset). Extbase/Fluid are considering both correctly for \DateTime properties.

What actually has to happen to solve the issue is that all time fields in the backend (no matter what concrete "eval") always store times as UTC timestamps in the database and treat them as such when displaying them in the backend. An upgrade wizard has to be shipped to convert all "eval = time" fields at the same time.

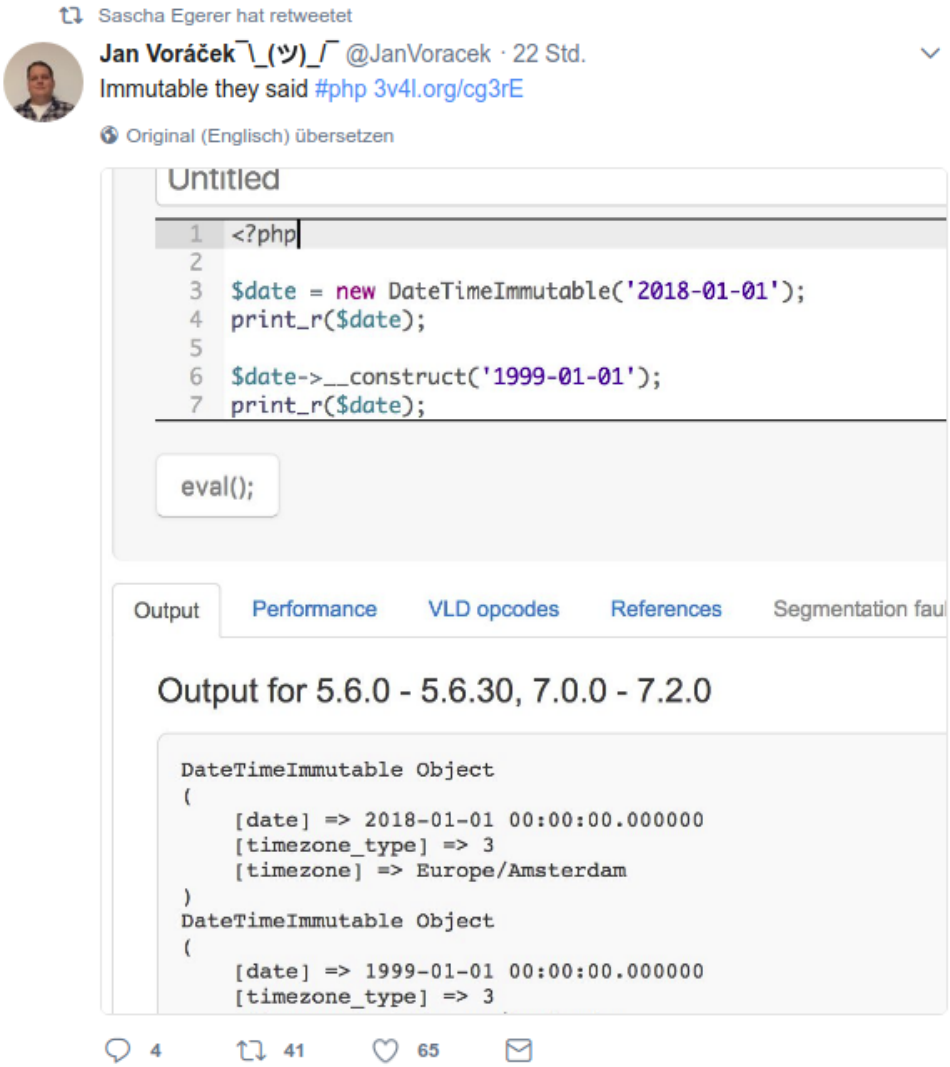
#33 - 2017-12-15 14:39 - Martin Bless

- File 330.png added

Additional information.

Just saw this:

https://3v4l.org/cg3rE



https://twitter.com/JanVoracek/status/941335582868934656

#34 - 2019-04-17 23:40 - Benni Mack

- Target version changed from next-patchlevel to Candidate for patchlevel

#35 - 2020-06-18 00:00 - Guillaume Crico

Having UTC timestamp in the database can be useful (ex: "timezone independent sorting", changing the PHP default timezone).

I think the bug is in TYPO3\CMS\Extbase\Persistence\Generic\Mapper\DataMapper->mapDateTime().

The following lines discard timezone information:

```
        // integer timestamps are local server time
        return \TYPO3\CMS\Core\Utility\GeneralUtility::makeInstance($targetType, date('c', (int)$value));
```

The expression "date('c', (int)$value)" returns a string with the form "2020-06-17T17:45:53-04:00", where the timezone offset is present (and results int the infamous "timezone_type=1"), but not the real timezone identifier that is needed to handle daylight saving.

Theses lines should be replaced by:

```
        // integer timestamps are regular (UTC) timestamps
        $datetime = new \Datetime('@' . (int) $value);
        $datetime->setTimezone(new \DateTimeZone(date_default_timezone_get()));
        return $datetime;
```

With this patch, the \Datetime properties have the proper timezone, that allows regular manipulations.

There is no need to "upgrade" existing data.

POC: https://3v4l.org/Y51mk

#36 - 2020-06-18 12:20 - Martin Bless

That really sounds like a substantial answer. I'm hoping somebody can verify this and finally get it implemented.

What an odyssey since 2015!

#37 - 2020-06-18 12:29 - Paul Golmann

Martin Bless wrote:

> That really sounds like a substantial answer. I'm hoping somebody can verify this and finally get it implemented.

Not to shamelessly promote myself, but this is exactly what I proposed in March 2017 in
https://forge.typo3.org/issues/80349 and
https://review.typo3.org/c/Packages/TYPO3.CMS/+/52952

but it may need an update for current core version.

#38 - 2021-09-22 16:24 - Hauke Loebert

For db datetime fields i see the major problem in the backend editor. The extbase

```
DataMapper->mapDateTime()
```

maps the db field (utc) via date_default_timezone_get() to the actual timezone. But the same value in the typo3 **backend is not translated** at all. So the editor see the utc time from db. What a mess.

Files

DateViewHelper.php	6.26 KB	2016-12-09	Jörg Velletti
330.png	144 KB	2017-12-15	Martin Bless