When building an Api using Symfony3, you'll be smart to process all input through `Forms`, and my strong recommendation is to do so via unique form classes for each Entity.

However, when doing so with custom `datetime` fields, you'll need to explicitly specify a format so you don't waste countless hours debugging what you are **certain** is a proper DateTime!

Using phpunit to validate a simple `node` creation, the entire request-to-response flow is as follows:

1. Api Request
2. Controller
3. Form
    1. Validation
4. ORM (Doctrine)
5. Response

This simple Unit Test is simulating a form submission with a `body` and `post_date` field populated for a simple Node entity. If we were creating this in a Controller or elsewhere, we could assign the value of `post_date` with a simple `DateTime` object and Symfony would do its magic of parsing it correctly for us.

For example, in the constructor of the `Node` you'll see I'm automatically assigning the `create_date` when the `Node` is first created.

```
public function __construct()
{
    $now = new \DateTime();
    $tz = new \DateTimeZone('America/New_York');
    $now->setTimezone($tz);
    $this->create_date = $now;
}
```

Unfortunately, when submitting to an Api, the object translation doesn't happen in a Unit Test as its passing raw encoded (JSON in this case) data. So we need to explicitly enter the date as a string with format that matches exactly what we’re expecting.

```
//tests/Api/Controllers/Node
public function testNodeCreate()
{
    $token = $this->authorize();

    $faker = \Faker\Factory::create('en_EN');

    $data = array(
        'body' => 'Fake description.',
        'post_date' => $faker->dateTimeThisMonth->format('Y-m-d
h:i:s'),
    );
    $url = 'http://api.myapp.dev/api/v1/nodes';

    $client = new Client($url, array(
        'request.options' => array(
            'exceptions' => false,
        )
    ));

    $request = $client->post($url, null, json_encode($data));
    $request->addHeader('Authorization', sprintf('Bearer %s',
$token));
    $response = $request->send();

    $this->assertEquals(200, $response->getStatusCode());

}
```

The form itself then simply needs to define the `post_date` field, the `widget` type and `format` allowed.

```
//MyApp/Bundle/CoreBundle/Forms/Node
use Symfony\Component\Form\Extension\Core\Type\DateTimeType;

public function buildForm(FormBuilderInterface $builder, array
$options)
{
    $builder
        ->add('body')
        ->add('post_date', DateTimeType::class, array(
            'invalid_message' => 'Please enter a valid date with time
for post date: Y-m-d h:i:s',
            'widget' => 'single_text',
            'format' => 'Y-m-d h:i:s',
        ))
        ->add('submit', SubmitType::class)
    ;
}
```

What’s important is that you match the input in your submission *precisely* to the format specified in the form. Submitting a DateTime` object will not be recognized correctly, nor will any string that does not explicitly match.