For: Dev Notes 6.4

Status: final

Dev Note for 20564

Framework for storing revisions of Post Meta in 6.4

Revisions are now supported for post meta on an opt-in basis. This feature is currently used by core for footnotes - enabling footnotes to be correctly previewed, stored in autosaved and stored and retrieved from revisions. In the future, this may be expanded to the featured image which is also stored in meta.

When registering a new meta field with `register_meta` or `register_post_meta`, the `$args` parameter accepts a new `revisions_enabled` argument. This defaults to false—set to true to opt in to revisions for this meta field. Note that the object type the meta is being created for must have revision support in order for the meta field to have revisions enabled. Attempting to enable meta revisions for a post type that doesn't support revisions will result in a `doing_it_wrong` warning.

This feature is designed to lay the groundwork for future improvements to Gutenberg in Phase 3: Collaboration. Care has been made to do this in a backwards compatible manner, but **if you are unhooking anything in the post save/update process, please read this entire note and test your code before WordPress 6.4 is released**.

A new `wp_post_revision_meta_keys` filter is introduced, which can be used to filter the list of meta fields to be revisioned. The post type is also passed to the filter, which can be used to adjust revisioning of meta fields.

```
// Example to add a `my_product_price` meta field to revisions.
function enable_revisions_for_product_price_field( $revisioned_keys ) {
	if ( ! in_array( 'my_product_price', $revisioned_keys, true ) ) {
		$revisioned_keys[] = 'product_price';
	}
	return $revisioned_keys;
}
add_filter( 'wp_post_revision_meta_keys', 'enable_revisions_for_product_price_field' );
```

The `_wp_put_post_revision` action, which fires once a revision is saved, includes a new parameter—the post ID associated with the revision. This is used by core to copy the meta data from the main post to the revision.

The `wp_creating_autosave` action, which fires before an autosave is stored, includes a new parameter `$is_update`, added to indicate if the autosave is being updated or was newly created.

Meta revisions are also stored for autosaves and when previewing posts, fixing a long standing bug (#20299) that caused live post meta to be overwritten when post meta was previewed. This is because using `update_post_meta` to save meta to a revision actually resulted in it being saved to the parent post. With this change, developers can now opt in to meta revisions and core will correctly attach the meta data to the revision. Meta that has revisions is also restored when restoring an autosave or revision.

The REST API revisions endpoint now returns meta when stored as part of a revision and the autosaves endpoint now handles storing of meta.

The new meta revision features are enabled in core via hooks; removing the hooks will disable the features:

To disable post meta revisions entirely, use:
`remove_action( 'wp_after_insert_post', 'wp_save_post_revision_on_insert', 9, 3 );`

To disable storing post meta on autosaves, use:
`remove_action( 'wp_creating_autosave', 'wp_autosave_post_revisioned_meta_fields' );`

To disable restoring meta when restoring revisions or autosaves, use:
`remove_action( 'wp_restore_post_revision', 'wp_restore_post_revision_meta', 10, 2 );`

Important note: as part of this change, the timing for the storing of post revisions has been changed. Previously, revisions were created before post meta had been saved by the REST API on the `post_updated` hook. Revisions are now hooked on the `wp_after_insert_post` action, fired after post meta has been saved.

Special backwards compatibility consideration has been given to plugins that may be unhooking `post_updated` from `wp_save_post_revision` to disable revisions—in that case, core ensures revisions are still not created.

Props to @westonruter, @jorbin and @webcommsat for reviewing.