Content Filters

The **Content Filters** in Replyke allow developers to filter entities based on the content of their `content` property. This functionality is versatile and can be used to include or exclude entities with specific content characteristics. Let's explore how Content Filters work and how to use them effectively.

Overview of Content Filters

The `contentFilters` property can be passed into the `EntityListProvider` as an object with the following structure:

```
export interface ContentFilters {
  hasContent?: boolean;
  includes?: string | string[];
  doesNotInclude?: string | string[];
}
```

- **`hasContent`**: A boolean flag to include entities based on whether they have content (`true`) or not (`false`).
- **`includes`**: A string or array of strings. Entities whose `content` contains any of these strings will be included.
- **`doesNotInclude`**: A string or array of strings. Entities whose `content` contains any of these strings will be excluded.

Dynamically Updating Content Filters

Developers can dynamically update the content filters using the `setContentFilters` function provided by the `useEntityList` hook. This function expects a new `contentFilters` object or `null` to clear the filters.

```
const { setContentFilters } = useEntityList();

// Set a new content filter
setContentFilters({ includes: "JavaScript", doesNotInclude: "PHP" });

// Reset content filters
setContentFilters(null);
```

How Content Filters Work

Content Filters are applied on the backend when querying entities. They allow for:

1. **Existence Filtering**: Using the `hasContent` flag, you can include only entities that have content (`true`) or exclude entities without content (`false`).
2. **Text Matching**:
   - `includes`: Filters entities to include those whose content contains any of the specified strings.
   - `doesNotInclude`: Filters entities to exclude those whose content contains any of the specified strings.
3. **Combined Conditions**: Developers can combine these conditions to achieve granular control over the results.

Example Use Cases

1. Passing Static Filters to the EntityListProvider

To set static filters for content at the initialization of the feed, pass them directly to the `EntityListProvider`:

```
<EntityListProvider
  contentFilters={{ hasContent: true, includes: ["JavaScript", "React"], doesNotInclude:
>
  <MyFeedComponent />
</EntityListProvider>
```

This setup ensures that only entities with content are included, and that content must contain either “JavaScript” or “React” while excluding “PHP.”

2. Dynamically Updating Filters Based on User Interaction

You can also dynamically adjust the content filters based on user actions, such as clicking a button:

```
const { setContentFilters, entities, loadMore } = useEntityList();

const handleFilterUpdate = (newIncludes: string[], newExcludes: string[]) => {
  setContentFilters({ includes: newIncludes, doesNotInclude: newExcludes });
};

return (
```

```
  <div>
    <h1>Filtered Content</h1>
    <div>
      <button onClick={() => handleFilterUpdate(["JavaScript"], ["PHP"])}>
        Show JavaScript, Exclude PHP
      </button>
      <button onClick={() => handleFilterUpdate(null)}>Clear Filters</button>
    </div>
    <ul>
      {entities.map((entity) => (
        <li key={entity.id}>
          <h2>{entity.title}</h2>
          <p>{entity.content}</p>
        </li>
      ))}
    </ul>
    <button onClick={loadMore}>Load More</button>
  </div>
);
```

In this example, clicking the button updates the content filters dynamically, allowing users to customize the feed results.

Important Notes

- Content Filters are case-insensitive, making them flexible for user-defined queries.
- Filters are applied immediately, and the feed resets to reflect the updated conditions.
- Passing `null` to `setContentFilters` clears all content-based filtering.

Conclusion

Content Filters provide developers with powerful tools to fine-tune feed results based on the content of entity properties. Whether setting static filters at initialization or dynamically updating them based on user input, Content Filters ensure your feed presents the most relevant content. With options to filter by existence, inclusion, and exclusion, you can cater to diverse application needs effortlessly.

Last updated on May 7, 2025