Replace AnimatableValues with ComputedStyle references

Owner: alancutter@
Last updated: 23rd May 2017
Tracking bug: https://crbug.com/725385

Objective

Minimise the conceptual overhead required for Blink developers to implement animations for new CSS properties.

Background

- AnimatableValues are a legacy implementation of keyframe effects in the animation engine that precedes our current use of Interpolations and InterpolationTypes.
- Despite being deprecated legacy code AnimatableValues are still used for reversing CSS Transitions and constructing compositor animation keyframes.
- Every time a new CSS property gets animated the developer needs to extend the AnimatableValue code further, sometimes even by adding new AnimatableValue subclasses. Example.
- “AnimatableValue” is not a good name for something that isn’t intended to animate its value. It was an accurate name a long time ago but not today.

Implementation Plan

Replace uses of AnimatableValues with ComputedStyles

- Store entire ComputedStyle object and CSSPropertyID pairs instead of storing individual property values represented by AnimatableValues.

Pros

- AnimatableValues are no longer a dependency of the animation engine.
- 17 out of 21 AnimatableValue subclasses will be deleted.
- AnimatableValue equality logic will be removed.
- Newly animated properties don’t need to deal with AnimatableValues.
- The binary size of libblink_core.so on Linux is reduced by ~48KB.

Cons

- Transitions need to clone ComputedStyle objects whenever they start.

- The previous ComputedStyle object will be kept alive during transitions.

Alternative Considered

Rename AnimatableValue to ComputedValue and merge with Clockwork project

- AnimatableValue gets renamed to ComputedValue.
- Eventually Clockwork will generate these ComputedValue classes.
- Legacy interpolation code for AnimatableValues will be moved to a static CompositorAnimations function.

Pros

- ComputedValue subclasses will eventually be auto generated by Clockwork.

Cons

- This adds to the complexity of the ComputedStyle generator and potentially all new CSS properties in general.

This option was discarded for its increased code complexity, binary size and long term maintenance costs compared to the chosen implementation plan.

Implementation CLs

Replace AnimatableValues with ComputedStyle references for CSS Transitions
Remove unused AnimatableValue types