4:39 PM 1/27/2024
1706390143304
B96555A0-2CCF-4BEB-BFC9-F27459B4AEA1

```
(function () {
    "use strict";

    var MessageTypeUItoBG;
    (function (MessageTypeUItoBG) {
        MessageTypeUItoBG["GET_DATA"] = "ui-bg-get-data";
        MessageTypeUItoBG["GET_DEVTOOLS_DATA"] = "ui-bg-get-devtools-data";
        MessageTypeUItoBG["SUBSCRIBE_TO_CHANGES"] =
            "ui-bg-subscribe-to-changes";
        MessageTypeUItoBG["UNSUBSCRIBE_FROM_CHANGES"] =
            "ui-bg-unsubscribe-from-changes";
        MessageTypeUItoBG["CHANGE_SETTINGS"] = "ui-bg-change-settings";
        MessageTypeUItoBG["SET_THEME"] = "ui-bg-set-theme";
        MessageTypeUItoBG["TOGGLE_ACTIVE_TAB"] = "ui-bg-toggle-active-tab";
        MessageTypeUItoBG["MARK_NEWS_AS_READ"] = "ui-bg-mark-news-as-read";
        MessageTypeUItoBG["MARK_NEWS_AS_DISPLAYED"] =
            "ui-bg-mark-news-as-displayed";
        MessageTypeUItoBG["LOAD_CONFIG"] = "ui-bg-load-config";
        MessageTypeUItoBG["APPLY_DEV_DYNAMIC_THEME_FIXES"] =
            "ui-bg-apply-dev-dynamic-theme-fixes";
        MessageTypeUItoBG["RESET_DEV_DYNAMIC_THEME_FIXES"] =
            "ui-bg-reset-dev-dynamic-theme-fixes";
        MessageTypeUItoBG["APPLY_DEV_INVERSION_FIXES"] =
            "ui-bg-apply-dev-inversion-fixes";
        MessageTypeUItoBG["RESET_DEV_INVERSION_FIXES"] =
            "ui-bg-reset-dev-inversion-fixes";
        MessageTypeUItoBG["APPLY_DEV_STATIC_THEMES"] =
            "ui-bg-apply-dev-static-themes";
        MessageTypeUItoBG["RESET_DEV_STATIC_THEMES"] =
            "ui-bg-reset-dev-static-themes";
        MessageTypeUItoBG["COLOR_SCHEME_CHANGE"] = "ui-bg-color-scheme-change";
        MessageTypeUItoBG["HIDE_HIGHLIGHTS"] = "ui-bg-hide-highlights";
    })(MessageTypeUItoBG || (MessageTypeUItoBG = {}));
    var MessageTypeBGtoUI;
    (function (MessageTypeBGtoUI) {
        MessageTypeBGtoUI["CHANGES"] = "bg-ui-changes";
    })(MessageTypeBGtoUI || (MessageTypeBGtoUI = {}));
    var DebugMessageTypeBGtoUI;
    (function (DebugMessageTypeBGtoUI) {
        DebugMessageTypeBGtoUI["CSS_UPDATE"] = "debug-bg-ui-css-update";
        DebugMessageTypeBGtoUI["UPDATE"] = "debug-bg-ui-update";
    })(DebugMessageTypeBGtoUI || (DebugMessageTypeBGtoUI = {}));
    var MessageTypeBGtoCS;
    (function (MessageTypeBGtoCS) {
        MessageTypeBGtoCS["ADD_CSS_FILTER"] = "bg-cs-add-css-filter";
        MessageTypeBGtoCS["ADD_DYNAMIC_THEME"] = "bg-cs-add-dynamic-theme";
        MessageTypeBGtoCS["ADD_STATIC_THEME"] = "bg-cs-add-static-theme";
        MessageTypeBGtoCS["ADD_SVG_FILTER"] = "bg-cs-add-svg-filter";
        MessageTypeBGtoCS["CLEAN_UP"] = "bg-cs-clean-up";
        MessageTypeBGtoCS["FETCH_RESPONSE"] = "bg-cs-fetch-response";
        MessageTypeBGtoCS["UNSUPPORTED_SENDER"] = "bg-cs-unsupported-sender";
    })(MessageTypeBGtoCS || (MessageTypeBGtoCS = {}));
    var DebugMessageTypeBGtoCS;
    (function (DebugMessageTypeBGtoCS) {
        DebugMessageTypeBGtoCS["RELOAD"] = "debug-bg-cs-reload";
    })(DebugMessageTypeBGtoCS || (DebugMessageTypeBGtoCS = {}));
    var MessageTypeCStoBG;
    (function (MessageTypeCStoBG) {
        MessageTypeCStoBG["COLOR_SCHEME_CHANGE"] = "cs-bg-color-scheme-change";
        MessageTypeCStoBG["DARK_THEME_DETECTED"] = "cs-bg-dark-theme-detected";
        MessageTypeCStoBG["DARK_THEME_NOT_DETECTED"] =
            "cs-bg-dark-theme-not-detected";
        MessageTypeCStoBG["FETCH"] = "cs-bg-fetch";
        MessageTypeCStoBG["DOCUMENT_CONNECT"] = "cs-bg-document-connect";
        MessageTypeCStoBG["DOCUMENT_FORGET"] = "cs-bg-document-forget";
```

```
    MessageTypeCStoBG["DOCUMENT_FREEZE"] = "cs-bg-document-freeze";
    MessageTypeCStoBG["DOCUMENT_RESUME"] = "cs-bg-document-resume";
})(MessageTypeCStoBG || (MessageTypeCStoBG = {}));
var DebugMessageTypeCStoBG;
(function (DebugMessageTypeCStoBG) {
    DebugMessageTypeCStoBG["LOG"] = "debug-cs-bg-log";
})(DebugMessageTypeCStoBG || (DebugMessageTypeCStoBG = {}));
var MessageTypeCStoUI;
(function (MessageTypeCStoUI) {
    MessageTypeCStoUI["EXPORT_CSS_RESPONSE"] = "cs-ui-export-css-response";
})(MessageTypeCStoUI || (MessageTypeCStoUI = {}));
var MessageTypeUItoCS;
(function (MessageTypeUItoCS) {
    MessageTypeUItoCS["EXPORT_CSS"] = "ui-cs-export-css";
})(MessageTypeUItoCS || (MessageTypeUItoCS = {}));

function logInfo(...args) {}
function logWarn(...args) {}
function logInfoCollapsed(title, ...args) {}

function throttle(callback) {
    let pending = false;
    let frameId = null;
    let lastArgs;
    const throttled = (...args) => {
        lastArgs = args;
        if (frameId) {
            pending = true;
        } else {
            callback(...lastArgs);
            frameId = requestAnimationFrame(() => {
                frameId = null;
                if (pending) {
                    callback(...lastArgs);
                    pending = false;
                }
            });
        }
    };
    const cancel = () => {
        cancelAnimationFrame(frameId);
        pending = false;
        frameId = null;
    };
    return Object.assign(throttled, {cancel});
}
function createAsyncTasksQueue() {
    const tasks = [];
    let frameId = null;
    function runTasks() {
        let task;
        while ((task = tasks.shift())) {
            task();
        }
        frameId = null;
    }
    function add(task) {
        tasks.push(task);
        if (!frameId) {
            frameId = requestAnimationFrame(runTasks);
        }
    }
    function cancel() {
        tasks.splice(0);
        cancelAnimationFrame(frameId);
        frameId = null;
    }
    return {add, cancel};
}
```

```
function isArrayLike(items) {
    return items.length != null;
}
function forEach(items, iterator) {
    if (isArrayLike(items)) {
        for (let i = 0, len = items.length; i < len; i++) {
            iterator(items[i]);
        }
    } else {
        for (const item of items) {
            iterator(item);
        }
    }
}
function push(array, addition) {
    forEach(addition, (a) => array.push(a));
}
function toArray(items) {
    const results = [];
    for (let i = 0, len = items.length; i < len; i++) {
        results.push(items[i]);
    }
    return results;
}

function getDuration(time) {
    let duration = 0;
    if (time.seconds) {
        duration += time.seconds * 1000;
    }
    if (time.minutes) {
        duration += time.minutes * 60 * 1000;
    }
    if (time.hours) {
        duration += time.hours * 60 * 60 * 1000;
    }
    if (time.days) {
        duration += time.days * 24 * 60 * 60 * 1000;
    }
    return duration;
}

function createNodeAsap({
    selectNode,
    createNode,
    updateNode,
    selectTarget,
    createTarget,
    isTargetMutation
}) {
    const target = selectTarget();
    if (target) {
        const prev = selectNode();
        if (prev) {
            updateNode(prev);
        } else {
            createNode(target);
        }
    } else {
        const observer = new MutationObserver((mutations) => {
            const mutation = mutations.find(isTargetMutation);
            if (mutation) {
                unsubscribe();
                const target = selectTarget();
                selectNode() || createNode(target);
            }
        });
        const ready = () => {
```

```
            if (document.readyState !== "complete") {
                return;
            }
            unsubscribe();
            const target = selectTarget() || createTarget();
            selectNode() || createNode(target);
        };
        const unsubscribe = () => {
            document.removeEventListener("readystatechange", ready);
            observer.disconnect();
        };
        if (document.readyState === "complete") {
            ready();
        } else {
            document.addEventListener("readystatechange", ready);
            observer.observe(document, {childList: true, subtree: true});
        }
    }
}
function removeNode(node) {
    node && node.parentNode && node.parentNode.removeChild(node);
}
function watchForNodePosition(node, mode, onRestore = Function.prototype) {
    const MAX_ATTEMPTS_COUNT = 10;
    const RETRY_TIMEOUT = getDuration({seconds: 2});
    const ATTEMPTS_INTERVAL = getDuration({seconds: 10});
    const prevSibling = node.previousSibling;
    let parent = node.parentNode;
    if (!parent) {
        throw new Error(
            "Unable to watch for node position: parent element not found"
        );
    }
    if (mode === "prev-sibling" && !prevSibling) {
        throw new Error(
            "Unable to watch for node position: there is no previous sibling"
        );
    }
    let attempts = 0;
    let start = null;
    let timeoutId = null;
    const restore = throttle(() => {
        if (timeoutId) {
            return;
        }
        attempts++;
        const now = Date.now();
        if (start == null) {
            start = now;
        } else if (attempts >= MAX_ATTEMPTS_COUNT) {
            if (now - start < ATTEMPTS_INTERVAL) {
                timeoutId = setTimeout(() => {
                    start = null;
                    attempts = 0;
                    timeoutId = null;
                    restore();
                }, RETRY_TIMEOUT);
                return;
            }
            start = now;
            attempts = 1;
        }
        if (mode === "head") {
            if (prevSibling && prevSibling.parentNode !== parent) {
                stop();
                return;
            }
        }
        if (mode === "prev-sibling") {
```

```
            if (prevSibling.parentNode == null) {
                stop();
                return;
            }
            if (prevSibling.parentNode !== parent) {
                updateParent(prevSibling.parentNode);
            }
        }
        if (mode === "head" && !parent.isConnected) {
            parent = document.head;
        }
        parent.insertBefore(
            node,
            prevSibling && prevSibling.isConnected
                ? prevSibling.nextSibling
                : parent.firstChild
        );
        observer.takeRecords();
        onRestore && onRestore();
    });
    const observer = new MutationObserver(() => {
        if (
            (mode === "head" &&
                (node.parentNode !== parent ||
                    !node.parentNode.isConnected)) ||
            (mode === "prev-sibling" &&
                node.previousSibling !== prevSibling)
        ) {
            restore();
        }
    });
    const run = () => {
        observer.observe(parent, {childList: true});
    };
    const stop = () => {
        clearTimeout(timeoutId);
        observer.disconnect();
        restore.cancel();
    };
    const skip = () => {
        observer.takeRecords();
    };
    const updateParent = (parentNode) => {
        parent = parentNode;
        stop();
        run();
    };
    run();
    return {run, stop, skip};
}
function iterateShadowHosts(root, iterator) {
    if (root == null) {
        return;
    }
    const walker = document.createTreeWalker(
        root,
        NodeFilter.SHOW_ELEMENT,
        {
            acceptNode(node) {
                return node.shadowRoot == null
                    ? NodeFilter.FILTER_SKIP
                    : NodeFilter.FILTER_ACCEPT;
            }
        }
    );
    for (
        let node = root.shadowRoot ? walker.currentNode : walker.nextNode();
        node != null;
        node = walker.nextNode()
```

```
    ) {
        if (node.classList.contains("surfingkeys_hints_host")) {
            continue;
        }
        iterator(node);
        iterateShadowHosts(node.shadowRoot, iterator);
    }
}
let isDOMReady = () => {
    return (
        document.readyState === "complete" ||
        document.readyState === "interactive"
    );
};
function setIsDOMReady(newFunc) {
    isDOMReady = newFunc;
}
const readyStateListeners = new Set();
function addDOMReadyListener(listener) {
    isDOMReady() ? listener() : readyStateListeners.add(listener);
}
function removeDOMReadyListener(listener) {
    readyStateListeners.delete(listener);
}
function isReadyStateComplete() {
    return document.readyState === "complete";
}
const readyStateCompleteListeners = new Set();
function addReadyStateCompleteListener(listener) {
    isReadyStateComplete()
        ? listener()
        : readyStateCompleteListeners.add(listener);
}
function cleanReadyStateCompleteListeners() {
    readyStateCompleteListeners.clear();
}
if (!isDOMReady()) {
    const onReadyStateChange = () => {
        if (isDOMReady()) {
            readyStateListeners.forEach((listener) => listener());
            readyStateListeners.clear();
            if (isReadyStateComplete()) {
                document.removeEventListener(
                    "readystatechange",
                    onReadyStateChange
                );
                readyStateCompleteListeners.forEach((listener) =>
                    listener()
                );
                readyStateCompleteListeners.clear();
            }
        }
    };
    document.addEventListener("readystatechange", onReadyStateChange);
}
const HUGE_MUTATIONS_COUNT = 1000;
function isHugeMutation(mutations) {
    if (mutations.length > HUGE_MUTATIONS_COUNT) {
        return true;
    }
    let addedNodesCount = 0;
    for (let i = 0; i < mutations.length; i++) {
        addedNodesCount += mutations[i].addedNodes.length;
        if (addedNodesCount > HUGE_MUTATIONS_COUNT) {
            return true;
        }
    }
    return false;
}
```

```
function getElementsTreeOperations(mutations) {
    const additions = new Set();
    const deletions = new Set();
    const moves = new Set();
    mutations.forEach((m) => {
        forEach(m.addedNodes, (n) => {
            if (n instanceof Element && n.isConnected) {
                additions.add(n);
            }
        });
        forEach(m.removedNodes, (n) => {
            if (n instanceof Element) {
                if (n.isConnected) {
                    moves.add(n);
                    additions.delete(n);
                } else {
                    deletions.add(n);
                }
            }
        });
    });
    const duplicateAdditions = [];
    const duplicateDeletions = [];
    additions.forEach((node) => {
        if (additions.has(node.parentElement)) {
            duplicateAdditions.push(node);
        }
    });
    deletions.forEach((node) => {
        if (deletions.has(node.parentElement)) {
            duplicateDeletions.push(node);
        }
    });
    duplicateAdditions.forEach((node) => additions.delete(node));
    duplicateDeletions.forEach((node) => deletions.delete(node));
    return {additions, moves, deletions};
}
const optimizedTreeObservers = new Map();
const optimizedTreeCallbacks = new WeakMap();
function createOptimizedTreeObserver(root, callbacks) {
    let observer;
    let observerCallbacks;
    let domReadyListener;
    if (optimizedTreeObservers.has(root)) {
        observer = optimizedTreeObservers.get(root);
        observerCallbacks = optimizedTreeCallbacks.get(observer);
    } else {
        let hadHugeMutationsBefore = false;
        let subscribedForReadyState = false;
        observer = new MutationObserver((mutations) => {
            if (isHugeMutation(mutations)) {
                if (!hadHugeMutationsBefore || isDOMReady()) {
                    observerCallbacks.forEach(({onHugeMutations}) =>
                        onHugeMutations(root)
                    );
                } else if (!subscribedForReadyState) {
                    domReadyListener = () =>
                        observerCallbacks.forEach(({onHugeMutations}) =>
                            onHugeMutations(root)
                        );
                    addDOMReadyListener(domReadyListener);
                    subscribedForReadyState = true;
                }
                hadHugeMutationsBefore = true;
            } else {
                const elementsOperations =
                    getElementsTreeOperations(mutations);
                observerCallbacks.forEach(({onMinorMutations}) =>
                    onMinorMutations(elementsOperations)
```

```
                );
            }
        });
        observer.observe(root, {childList: true, subtree: true});
        optimizedTreeObservers.set(root, observer);
        observerCallbacks = new Set();
        optimizedTreeCallbacks.set(observer, observerCallbacks);
    }
    observerCallbacks.add(callbacks);
    return {
        disconnect() {
            observerCallbacks.delete(callbacks);
            if (domReadyListener) {
                removeDOMReadyListener(domReadyListener);
            }
            if (observerCallbacks.size === 0) {
                observer.disconnect();
                optimizedTreeCallbacks.delete(observer);
                optimizedTreeObservers.delete(root);
            }
        }
    };
}

function createOrUpdateStyle$1(css, type) {
    createNodeAsap({
        selectNode: () => document.getElementById("dark-reader-style"),
        createNode: (target) => {
            document.documentElement.setAttribute(
                "data-darkreader-mode",
                type
            );
            const style = document.createElement("style");
            style.id = "dark-reader-style";
            style.classList.add("darkreader");
            style.type = "text/css";
            style.textContent = css;
            target.appendChild(style);
        },
        updateNode: (existing) => {
            if (
                css.replace(/^\s+/gm, "") !==
                existing.textContent.replace(/^\s+/gm, "")
            ) {
                existing.textContent = css;
            }
        },
        selectTarget: () => document.head,
        createTarget: () => {
            const head = document.createElement("head");
            document.documentElement.insertBefore(
                head,
                document.documentElement.firstElementChild
            );
            return head;
        },
        isTargetMutation: (mutation) =>
            mutation.target.nodeName.toLowerCase() === "head"
    });
}
function removeStyle() {
    removeNode(document.getElementById("dark-reader-style"));
    document.documentElement.removeAttribute("data-darkreader-mode");
}

function createOrUpdateSVGFilter(svgMatrix, svgReverseMatrix) {
    createNodeAsap({
        selectNode: () => document.getElementById("dark-reader-svg"),
        createNode: (target) => {
```

```
            const SVG_NS = "http://www.w3.org/2000/svg";
            const createMatrixFilter = (id, matrix) => {
                const filter = document.createElementNS(SVG_NS, "filter");
                filter.id = id;
                filter.style.colorInterpolationFilters = "sRGB";
                filter.setAttribute("x", "0");
                filter.setAttribute("y", "0");
                filter.setAttribute("width", "99999");
                filter.setAttribute("height", "99999");
                filter.appendChild(createColorMatrix(matrix));
                return filter;
            };
            const createColorMatrix = (matrix) => {
                const colorMatrix = document.createElementNS(
                    SVG_NS,
                    "feColorMatrix"
                );
                colorMatrix.setAttribute("type", "matrix");
                colorMatrix.setAttribute("values", matrix);
                return colorMatrix;
            };
            const svg = document.createElementNS(SVG_NS, "svg");
            svg.id = "dark-reader-svg";
            svg.style.height = "0";
            svg.style.width = "0";
            svg.appendChild(
                createMatrixFilter("dark-reader-filter", svgMatrix)
            );
            svg.appendChild(
                createMatrixFilter(
                    "dark-reader-reverse-filter",
                    svgReverseMatrix
                )
            );
            target.appendChild(svg);
        },
        updateNode: (existing) => {
            const existingMatrix = existing.firstChild.firstChild;
            if (existingMatrix.getAttribute("values") !== svgMatrix) {
                existingMatrix.setAttribute("values", svgMatrix);
                const style = document.getElementById("dark-reader-style");
                const css = style.textContent;
                style.textContent = "";
                style.textContent = css;
            }
        },
        selectTarget: () => document.head,
        createTarget: () => {
            const head = document.createElement("head");
            document.documentElement.insertBefore(
                head,
                document.documentElement.firstElementChild
            );
            return head;
        },
        isTargetMutation: (mutation) =>
            mutation.target.nodeName.toLowerCase() === "head"
    });
}
function removeSVGFilter() {
    removeNode(document.getElementById("dark-reader-svg"));
}

function evalMath(expression) {
    const rpnStack = [];
    const workingStack = [];
    let lastToken;
    for (let i = 0, len = expression.length; i < len; i++) {
        const token = expression[i];
```

```
        if (!token || token === " ") {
            continue;
        }
        if (operators.has(token)) {
            const op = operators.get(token);
            while (workingStack.length) {
                const currentOp = operators.get(workingStack[0]);
                if (!currentOp) {
                    break;
                }
                if (op.lessOrEqualThan(currentOp)) {
                    rpnStack.push(workingStack.shift());
                } else {
                    break;
                }
            }
            workingStack.unshift(token);
        } else if (!lastToken || operators.has(lastToken)) {
            rpnStack.push(token);
        } else {
            rpnStack[rpnStack.length - 1] += token;
        }
        lastToken = token;
    }
    rpnStack.push(...workingStack);
    const stack = [];
    for (let i = 0, len = rpnStack.length; i < len; i++) {
        const op = operators.get(rpnStack[i]);
        if (op) {
            const args = stack.splice(0, 2);
            stack.push(op.exec(args[1], args[0]));
        } else {
            stack.unshift(parseFloat(rpnStack[i]));
        }
    }
    return stack[0];
}
class Operator {
    constructor(precedence, method) {
        this.precendce = precedence;
        this.execMethod = method;
    }
    exec(left, right) {
        return this.execMethod(left, right);
    }
    lessOrEqualThan(op) {
        return this.precendce <= op.precendce;
    }
}
const operators = new Map([
    ["+", new Operator(1, (left, right) => left + right)],
    ["-", new Operator(1, (left, right) => left - right)],
    ["*", new Operator(2, (left, right) => left * right)],
    ["/", new Operator(2, (left, right) => left / right)]
]);

function getMatches(regex, input, group = 0) {
    const matches = [];
    let m;
    while ((m = regex.exec(input))) {
        matches.push(m[group]);
    }
    return matches;
}
function formatCSS(text) {
    function trimLeft(text) {
        return text.replace(/^\s+/, "");
    }
    function getIndent(depth) {
```

```
        if (depth === 0) {
            return "";
        }
        return " ".repeat(4 * depth);
    }
    if (text.length < 50000) {
        const emptyRuleRegexp = /[^{}]+{\s*}/;
        while (emptyRuleRegexp.test(text)) {
            text = text.replace(emptyRuleRegexp, "");
        }
    }
    const css = text
        .replace(/\s{2,}/g, " ")
        .replace(/\{/g, "{\n")
        .replace(/\}/g, "\n}\n")
        .replace(/\;(?![^\(|\"]*(\)|\"))/g, ";\n")
        .replace(/\,(?![^\(|\"]*(\)|\"))/g, ",\n")
        .replace(/\n\s*\n/g, "\n")
        .split("\n");
    let depth = 0;
    const formatted = [];
    for (let x = 0, len = css.length; x < len; x++) {
        const line = `${css[x]}\n`;
        if (line.includes("{")) {
            formatted.push(getIndent(depth++) + trimLeft(line));
        } else if (line.includes("}")) {
            formatted.push(getIndent(--depth) + trimLeft(line));
        } else {
            formatted.push(getIndent(depth) + trimLeft(line));
        }
    }
    return formatted.join("").trim();
}
function getParenthesesRange(input, searchStartIndex = 0) {
    const length = input.length;
    let depth = 0;
    let firstOpenIndex = -1;
    for (let i = searchStartIndex; i < length; i++) {
        if (depth === 0) {
            const openIndex = input.indexOf("(", i);
            if (openIndex < 0) {
                break;
            }
            firstOpenIndex = openIndex;
            depth++;
            i = openIndex;
        } else {
            const closingIndex = input.indexOf(")", i);
            if (closingIndex < 0) {
                break;
            }
            const openIndex = input.indexOf("(", i);
            if (openIndex < 0 || closingIndex < openIndex) {
                depth--;
                if (depth === 0) {
                    return {start: firstOpenIndex, end: closingIndex + 1};
                }
                i = closingIndex;
            } else {
                depth++;
                i = openIndex;
            }
        }
    }
    return null;
}

const hslaParseCache = new Map();
const rgbaParseCache = new Map();
```

```
function parseColorWithCache($color) {
    $color = $color.trim();
    if (rgbaParseCache.has($color)) {
        return rgbaParseCache.get($color);
    }
    if ($color.includes("calc(")) {
        $color = lowerCalcExpression($color);
    }
    const color = parse($color);
    color && rgbaParseCache.set($color, color);
    return color;
}
function parseToHSLWithCache(color) {
    if (hslaParseCache.has(color)) {
        return hslaParseCache.get(color);
    }
    const rgb = parseColorWithCache(color);
    if (!rgb) {
        return null;
    }
    const hsl = rgbToHSL(rgb);
    hslaParseCache.set(color, hsl);
    return hsl;
}
function clearColorCache() {
    hslaParseCache.clear();
    rgbaParseCache.clear();
}
function hslToRGB({h, s, l, a = 1}) {
    if (s === 0) {
        const [r, b, g] = [l, l, l].map((x) => Math.round(x * 255));
        return {r, g, b, a};
    }
    const c = (1 - Math.abs(2 * l - 1)) * s;
    const x = c * (1 - Math.abs(((h / 60) % 2) - 1));
    const m = l - c / 2;
    const [r, g, b] = (
        h < 60
            ? [c, x, 0]
            : h < 120
              ? [x, c, 0]
              : h < 180
                ? [0, c, x]
                : h < 240
                  ? [0, x, c]
                  : h < 300
                    ? [x, 0, c]
                    : [c, 0, x]
    ).map((n) => Math.round((n + m) * 255));
    return {r, g, b, a};
}
function rgbToHSL({r: r255, g: g255, b: b255, a = 1}) {
    const r = r255 / 255;
    const g = g255 / 255;
    const b = b255 / 255;
    const max = Math.max(r, g, b);
    const min = Math.min(r, g, b);
    const c = max - min;
    const l = (max + min) / 2;
    if (c === 0) {
        return {h: 0, s: 0, l, a};
    }
    let h =
        (max === r
            ? ((g - b) / c) % 6
            : max === g
              ? (b - r) / c + 2
              : (r - g) / c + 4) * 60;
    if (h < 0) {
```

```
        h += 360;
    }
    const s = c / (1 - Math.abs(2 * l - 1));
    return {h, s, l, a};
}
function toFixed(n, digits = 0) {
    const fixed = n.toFixed(digits);
    if (digits === 0) {
        return fixed;
    }
    const dot = fixed.indexOf(".");
    if (dot >= 0) {
        const zerosMatch = fixed.match(/0+$/);
        if (zerosMatch) {
            if (zerosMatch.index === dot + 1) {
                return fixed.substring(0, dot);
            }
            return fixed.substring(0, zerosMatch.index);
        }
    }
    return fixed;
}
function rgbToString(rgb) {
    const {r, g, b, a} = rgb;
    if (a != null && a < 1) {
        return `rgba(${toFixed(r)}, ${toFixed(g)}, ${toFixed(b)}, ${toFixed(
            a,
            2
        )})`;
    }
    return `rgb(${toFixed(r)}, ${toFixed(g)}, ${toFixed(b)})`;
}
function rgbToHexString({r, g, b, a}) {
    return `#${(a != null && a < 1
        ? [r, g, b, Math.round(a * 255)]
        : [r, g, b]
    )
        .map((x) => {
            return `${x < 16 ? "0" : ""}${x.toString(16)}`;
        })
        .join("")}`;
}
function hslToString(hsl) {
    const {h, s, l, a} = hsl;
    if (a != null && a < 1) {
        return `hsla(${toFixed(h)}, ${toFixed(s * 100)}%, ${toFixed(
            l * 100
        )}%, ${toFixed(a, 2)})`;
    }
    return `hsl(${toFixed(h)}, ${toFixed(s * 100)}%, ${toFixed(l * 100)}%)`;
}
const rgbMatch = /^rgba?\([^\(\)]+\)$/;
const hslMatch = /^hsla?\([^\(\)]+\)$/;
const hexMatch = /^#[0-9a-f]+$/i;
function parse($color) {
    const c = $color.trim().toLowerCase();
    if (c.match(rgbMatch)) {
        return parseRGB(c);
    }
    if (c.match(hslMatch)) {
        return parseHSL(c);
    }
    if (c.match(hexMatch)) {
        return parseHex(c);
    }
    if (knownColors.has(c)) {
        return getColorByName(c);
    }
    if (systemColors.has(c)) {
```

```
        return getSystemColor(c);
    }
    if ($color === "transparent") {
        return {r: 0, g: 0, b: 0, a: 0};
    }
    return null;
}
function getNumbers($color) {
    const numbers = [];
    let prevPos = 0;
    let isMining = false;
    const startIndex = $color.indexOf("(");
    $color = $color.substring(startIndex + 1, $color.length - 1);
    for (let i = 0; i < $color.length; i++) {
        const c = $color[i];
        if ((c >= "0" && c <= "9") || c === "." || c === "+" || c === "-") {
            isMining = true;
        } else if (isMining && (c === " " || c === "," || c === "/")) {
            numbers.push($color.substring(prevPos, i));
            isMining = false;
            prevPos = i + 1;
        } else if (!isMining) {
            prevPos = i + 1;
        }
    }
    if (isMining) {
        numbers.push($color.substring(prevPos, $color.length));
    }
    return numbers;
}
function getNumbersFromString(str, range, units) {
    const raw = getNumbers(str);
    const unitsList = Object.entries(units);
    const numbers = raw
        .map((r) => r.trim())
        .map((r, i) => {
            let n;
            const unit = unitsList.find(([u]) => r.endsWith(u));
            if (unit) {
                n =
                    (parseFloat(r.substring(0, r.length - unit[0].length)) /
                        unit[1]) *
                    range[i];
            } else {
                n = parseFloat(r);
            }
            if (range[i] > 1) {
                return Math.round(n);
            }
            return n;
        });
    return numbers;
}
const rgbRange = [255, 255, 255, 1];
const rgbUnits = {"%": 100};
function parseRGB($rgb) {
    const [r, g, b, a = 1] = getNumbersFromString($rgb, rgbRange, rgbUnits);
    return {r, g, b, a};
}
const hslRange = [360, 1, 1, 1];
const hslUnits = {"%": 100, "deg": 360, "rad": 2 * Math.PI, "turn": 1};
function parseHSL($hsl) {
    const [h, s, l, a = 1] = getNumbersFromString($hsl, hslRange, hslUnits);
    return hslToRGB({h, s, l, a});
}
function parseHex($hex) {
    const h = $hex.substring(1);
    switch (h.length) {
        case 3:
```

```
        case 4: {
            const [r, g, b] = [0, 1, 2].map((i) =>
                parseInt(`${h[i]}${h[i]}`, 16)
            );
            const a =
                h.length === 3 ? 1 : parseInt(`${h[3]}${h[3]}`, 16) / 255;
            return {r, g, b, a};
        }
        case 6:
        case 8: {
            const [r, g, b] = [0, 2, 4].map((i) =>
                parseInt(h.substring(i, i + 2), 16)
            );
            const a =
                h.length === 6 ? 1 : parseInt(h.substring(6, 8), 16) / 255;
            return {r, g, b, a};
        }
    }
    return null;
}
function getColorByName($color) {
    const n = knownColors.get($color);
    return {
        r: (n >> 16) & 255,
        g: (n >> 8) & 255,
        b: (n >> 0) & 255,
        a: 1
    };
}
function getSystemColor($color) {
    const n = systemColors.get($color);
    return {
        r: (n >> 16) & 255,
        g: (n >> 8) & 255,
        b: (n >> 0) & 255,
        a: 1
    };
}
function lowerCalcExpression(color) {
    let searchIndex = 0;
    const replaceBetweenIndices = (start, end, replacement) => {
        color =
            color.substring(0, start) + replacement + color.substring(end);
    };
    while ((searchIndex = color.indexOf("calc(")) !== -1) {
        const range = getParenthesesRange(color, searchIndex);
        if (!range) {
            break;
        }
        let slice = color.slice(range.start + 1, range.end - 1);
        const includesPercentage = slice.includes("%");
        slice = slice.split("%").join("");
        const output = Math.round(evalMath(slice));
        replaceBetweenIndices(
            range.start - 4,
            range.end,
            output + (includesPercentage ? "%" : "")
        );
    }
    return color;
}
const knownColors = new Map(
    Object.entries({
        aliceblue: 0xf0f8ff,
        antiquewhite: 0xfaebd7,
        aqua: 0x00ffff,
        aquamarine: 0x7fffd4,
        azure: 0xf0ffff,
        beige: 0xf5f5dc,
```

```
bisque: 0xffe4c4,
black: 0x000000,
blanchedalmond: 0xffebcd,
blue: 0x0000ff,
blueviolet: 0x8a2be2,
brown: 0xa52a2a,
burlywood: 0xdeb887,
cadetblue: 0x5f9ea0,
chartreuse: 0x7fff00,
chocolate: 0xd2691e,
coral: 0xff7f50,
cornflowerblue: 0x6495ed,
cornsilk: 0xfff8dc,
crimson: 0xdc143c,
cyan: 0x00ffff,
darkblue: 0x00008b,
darkcyan: 0x008b8b,
darkgoldenrod: 0xb8860b,
darkgray: 0xa9a9a9,
darkgrey: 0xa9a9a9,
darkgreen: 0x006400,
darkkhaki: 0xbdb76b,
darkmagenta: 0x8b008b,
darkolivegreen: 0x556b2f,
darkorange: 0xff8c00,
darkorchid: 0x9932cc,
darkred: 0x8b0000,
darksalmon: 0xe9967a,
darkseagreen: 0x8fbc8f,
darkslateblue: 0x483d8b,
darkslategray: 0x2f4f4f,
darkslategrey: 0x2f4f4f,
darkturquoise: 0x00ced1,
darkviolet: 0x9400d3,
deeppink: 0xff1493,
deepskyblue: 0x00bfff,
dimgray: 0x696969,
dimgrey: 0x696969,
dodgerblue: 0x1e90ff,
firebrick: 0xb22222,
floralwhite: 0xfffaf0,
forestgreen: 0x228b22,
fuchsia: 0xff00ff,
gainsboro: 0xdcdcdc,
ghostwhite: 0xf8f8ff,
gold: 0xffd700,
goldenrod: 0xdaa520,
gray: 0x808080,
grey: 0x808080,
green: 0x008000,
greenyellow: 0xadff2f,
honeydew: 0xf0fff0,
hotpink: 0xff69b4,
indianred: 0xcd5c5c,
indigo: 0x4b0082,
ivory: 0xfffff0,
khaki: 0xf0e68c,
lavender: 0xe6e6fa,
lavenderblush: 0xfff0f5,
lawngreen: 0x7cfc00,
lemonchiffon: 0xfffacd,
lightblue: 0xadd8e6,
lightcoral: 0xf08080,
lightcyan: 0xe0ffff,
lightgoldenrodyellow: 0xfafad2,
lightgray: 0xd3d3d3,
lightgrey: 0xd3d3d3,
lightgreen: 0x90ee90,
lightpink: 0xffb6c1,
```

```
lightsalmon: 0xffa07a,
lightseagreen: 0x20b2aa,
lightskyblue: 0x87cefa,
lightslategray: 0x778899,
lightslategrey: 0x778899,
lightsteelblue: 0xb0c4de,
lightyellow: 0xffffe0,
lime: 0x00ff00,
limegreen: 0x32cd32,
linen: 0xfaf0e6,
magenta: 0xff00ff,
maroon: 0x800000,
mediumaquamarine: 0x66cdaa,
mediumblue: 0x0000cd,
mediumorchid: 0xba55d3,
mediumpurple: 0x9370db,
mediumseagreen: 0x3cb371,
mediumslateblue: 0x7b68ee,
mediumspringgreen: 0x00fa9a,
mediumturquoise: 0x48d1cc,
mediumvioletred: 0xc71585,
midnightblue: 0x191970,
mintcream: 0xf5fffa,
mistyrose: 0xffe4e1,
moccasin: 0xffe4b5,
navajowhite: 0xffdead,
navy: 0x000080,
oldlace: 0xfdf5e6,
olive: 0x808000,
olivedrab: 0x6b8e23,
orange: 0xffa500,
orangered: 0xff4500,
orchid: 0xda70d6,
palegoldenrod: 0xeee8aa,
palegreen: 0x98fb98,
paleturquoise: 0xafeeee,
palevioletred: 0xdb7093,
papayawhip: 0xffefd5,
peachpuff: 0xffdab9,
peru: 0xcd853f,
pink: 0xffc0cb,
plum: 0xdda0dd,
powderblue: 0xb0e0e6,
purple: 0x800080,
rebeccapurple: 0x663399,
red: 0xff0000,
rosybrown: 0xbc8f8f,
royalblue: 0x4169e1,
saddlebrown: 0x8b4513,
salmon: 0xfa8072,
sandybrown: 0xf4a460,
seagreen: 0x2e8b57,
seashell: 0xfff5ee,
sienna: 0xa0522d,
silver: 0xc0c0c0,
skyblue: 0x87ceeb,
slateblue: 0x6a5acd,
slategray: 0x708090,
slategrey: 0x708090,
snow: 0xfffafa,
springgreen: 0x00ff7f,
steelblue: 0x4682b4,
tan: 0xd2b48c,
teal: 0x008080,
thistle: 0xd8bfd8,
tomato: 0xff6347,
turquoise: 0x40e0d0,
violet: 0xee82ee,
wheat: 0xf5deb3,
```

```
        white: 0xffffff,
        whitesmoke: 0xf5f5f5,
        yellow: 0xffff00,
        yellowgreen: 0x9acd32
    })
);
const systemColors = new Map(
    Object.entries({
        "ActiveBorder": 0x3b99fc,
        "ActiveCaption": 0x000000,
        "AppWorkspace": 0xaaaaaa,
        "Background": 0x6363ce,
        "ButtonFace": 0xffffff,
        "ButtonHighlight": 0xe9e9e9,
        "ButtonShadow": 0x9fa09f,
        "ButtonText": 0x000000,
        "CaptionText": 0x000000,
        "GrayText": 0x7f7f7f,
        "Highlight": 0xb2d7ff,
        "HighlightText": 0x000000,
        "InactiveBorder": 0xffffff,
        "InactiveCaption": 0xffffff,
        "InactiveCaptionText": 0x000000,
        "InfoBackground": 0xfbfcc5,
        "InfoText": 0x000000,
        "Menu": 0xf6f6f6,
        "MenuText": 0xffffff,
        "Scrollbar": 0xaaaaaa,
        "ThreeDDarkShadow": 0x000000,
        "ThreeDFace": 0xc0c0c0,
        "ThreeDHighlight": 0xffffff,
        "ThreeDLightShadow": 0xffffff,
        "ThreeDShadow": 0x000000,
        "Window": 0xecececec,
        "WindowFrame": 0xaaaaaa,
        "WindowText": 0x000000,
        "-webkit-focus-ring-color": 0xe59700
    }).map(([key, value]) => [key.toLowerCase(), value])
);
function getSRGBLightness(r, g, b) {
    return (0.2126 * r + 0.7152 * g + 0.0722 * b) / 255;
}

const isNavigatorDefined = typeof navigator !== "undefined";
const userAgent = isNavigatorDefined
    ? navigator.userAgentData &&
      Array.isArray(navigator.userAgentData.brands)
        ? navigator.userAgentData.brands
              .map(
                  (brand) => `${brand.brand.toLowerCase()} ${brand.version}`
              )
              .join(" ")
        : navigator.userAgent.toLowerCase()
    : "some useragent";
const platform = isNavigatorDefined
    ? navigator.userAgentData &&
      typeof navigator.userAgentData.platform === "string"
        ? navigator.userAgentData.platform.toLowerCase()
        : navigator.platform.toLowerCase()
    : "some platform";
userAgent.includes("vivaldi");
userAgent.includes("yabrowser");
userAgent.includes("opr") || userAgent.includes("opera");
userAgent.includes("edg");
platform.startsWith("win");
platform.startsWith("mac");
isNavigatorDefined && navigator.userAgentData
    ? navigator.userAgentData.mobile
    : userAgent.includes("mobile");
```

```
const isShadowDomSupported = typeof ShadowRoot === "function";
const isMatchMediaChangeEventListenerSupported =
    typeof MediaQueryList === "function" &&
    typeof MediaQueryList.prototype.addEventListener === "function";
(isNavigatorDefined &&
    navigator.userAgentData &&
    ["Linux", "Android"].includes(navigator.userAgentData.platform)) ||
    platform.startsWith("linux");
(() => {
    const m = userAgent.match(/chrom(?:e|ium)(?:\/| )([^ ]+)/);
    if (m && m[1]) {
        return m[1];
    }
    return "";
})();
(() => {
    const m = userAgent.match(/(?:firefox|librewolf)(?:\/| )([^ ]+)/);
    if (m && m[1]) {
        return m[1];
    }
    return "";
})();
const isDefinedSelectorSupported = (() => {
    try {
        document.querySelector(":defined");
        return true;
    } catch (err) {
        return false;
    }
})();
const isCSSColorSchemePropSupported = (() => {
    try {
        if (typeof document === "undefined") {
            return false;
        }
        const el = document.createElement("div");
        if (!el || typeof el.style !== "object") {
            return false;
        }
        if (typeof el.style.colorScheme === "string") {
            return true;
        }
        el.setAttribute("style", "color-scheme: dark");
        return el.style.colorScheme === "dark";
    } catch (e) {
        return false;
    }
})();

let query = null;
const onChange = ({matches}) =>
    listeners.forEach((listener) => listener(matches));
const listeners = new Set();
function runColorSchemeChangeDetector(callback) {
    listeners.add(callback);
    if (query) {
        return;
    }
    query = matchMedia("(prefers-color-scheme: dark)");
    if (isMatchMediaChangeEventListenerSupported) {
        query.addEventListener("change", onChange);
    } else {
        query.addListener(onChange);
    }
}
function stopColorSchemeChangeDetector() {
    if (!query || !onChange) {
        return;
    }
```

```
    if (isMatchMediaChangeEventListenerSupported) {
        query.removeEventListener("change", onChange);
    } else {
        query.removeListener(onChange);
    }
    listeners.clear();
    query = null;
}
const isSystemDarkModeEnabled = () =>
    (query || matchMedia("(prefers-color-scheme: dark)")).matches;

const COLOR_SCHEME_META_SELECTOR = 'meta[name="color-scheme"]';
function hasBuiltInDarkTheme() {
    const rootStyle = getComputedStyle(document.documentElement);
    if (rootStyle.filter.includes("invert(1)")) {
        return true;
    }
    const CELL_SIZE = 256;
    const MAX_ROW_COUNT = 4;
    const winWidth = innerWidth;
    const winHeight = innerHeight;
    const stepX = Math.floor(
        winWidth / Math.min(MAX_ROW_COUNT, Math.ceil(winWidth / CELL_SIZE))
    );
    const stepY = Math.floor(
        winHeight /
            Math.min(MAX_ROW_COUNT, Math.ceil(winHeight / CELL_SIZE))
    );
    const processedElements = new Set();
    for (let y = Math.floor(stepY / 2); y < winHeight; y += stepY) {
        for (let x = Math.floor(stepX / 2); x < winWidth; x += stepX) {
            const element = document.elementFromPoint(x, y);
            if (!element || processedElements.has(element)) {
                continue;
            }
            processedElements.add(element);
            const style =
                element === document.documentElement
                    ? rootStyle
                    : getComputedStyle(element);
            const bgColor = parseColorWithCache(style.backgroundColor);
            if (bgColor.a === 1) {
                const bgLightness = getSRGBLightness(
                    bgColor.r,
                    bgColor.g,
                    bgColor.b
                );
                if (bgLightness > 0.5) {
                    return false;
                }
            } else {
                const textColor = parseColorWithCache(style.color);
                const textLightness = getSRGBLightness(
                    textColor.r,
                    textColor.g,
                    textColor.b
                );
                if (textLightness < 0.5) {
                    return false;
                }
            }
        }
    }
    const rootColor = parseColorWithCache(rootStyle.backgroundColor);
    const bodyColor = document.body
        ? parseColorWithCache(
              getComputedStyle(document.body).backgroundColor
          )
        : {r: 0, g: 0, b: 0, a: 0};
```

```
    const rootLightness =
        1 -
        rootColor.a +
        rootColor.a *
            getSRGBLightness(rootColor.r, rootColor.g, rootColor.b);
    const finalLightness =
        (1 - bodyColor.a) * rootLightness +
        bodyColor.a *
            getSRGBLightness(bodyColor.r, bodyColor.g, bodyColor.b);
    return finalLightness < 0.5;
}
function runCheck(callback) {
    const colorSchemeMeta = document.querySelector(
        COLOR_SCHEME_META_SELECTOR
    );
    if (colorSchemeMeta) {
        const isMetaDark =
            colorSchemeMeta.content === "dark" ||
            (colorSchemeMeta.content.includes("dark") &&
                isSystemDarkModeEnabled());
        callback(isMetaDark);
        return;
    }
    const drStyles = document.querySelectorAll(".darkreader");
    drStyles.forEach((style) => (style.disabled = true));
    const darkThemeDetected = hasBuiltInDarkTheme();
    drStyles.forEach((style) => (style.disabled = false));
    callback(darkThemeDetected);
}
function hasSomeStyle() {
    if (document.querySelector(COLOR_SCHEME_META_SELECTOR) != null) {
        return true;
    }
    if (
        document.documentElement.style.backgroundColor ||
        (document.body && document.body.style.backgroundColor)
    ) {
        return true;
    }
    for (const style of document.styleSheets) {
        if (
            style &&
            style.ownerNode &&
            !(
                style.ownerNode.classList &&
                style.ownerNode.classList.contains("darkreader")
            )
        ) {
            return true;
        }
    }
    return false;
}
let observer$1;
let readyStateListener;
function runDarkThemeDetector(callback, hints) {
    stopDarkThemeDetector();
    if (hints && hints.length > 0) {
        const hint = hints[0];
        if (hint.noDarkTheme) {
            callback(false);
            return;
        }
        if (hint.systemTheme && isSystemDarkModeEnabled()) {
            callback(true);
            return;
        }
        detectUsingHint(hint, () => callback(true));
        return;
```

```
    }
    if (document.body && hasSomeStyle()) {
        runCheck(callback);
        return;
    }
    observer$1 = new MutationObserver(() => {
        if (document.body && hasSomeStyle()) {
            stopDarkThemeDetector();
            runCheck(callback);
        }
    });
    observer$1.observe(document.documentElement, {childList: true});
    if (document.readyState !== "complete") {
        readyStateListener = () => {
            if (document.readyState === "complete") {
                stopDarkThemeDetector();
                runCheck(callback);
            }
        };
        document.addEventListener("readystatechange", readyStateListener);
    }
}
function stopDarkThemeDetector() {
    if (observer$1) {
        observer$1.disconnect();
        observer$1 = null;
    }
    if (readyStateListener) {
        document.removeEventListener(
            "readystatechange",
            readyStateListener
        );
        readyStateListener = null;
    }
    stopDetectingUsingHint();
}
let hintTargetObserver;
let hintMatchObserver;
function detectUsingHint(hint, success) {
    stopDetectingUsingHint();
    const matchSelector = (hint.match || []).join(", ");
    function checkMatch(target) {
        var _a;
        if (
            (_a = target.matches) === null || _a === void 0
                ? void 0
                : _a.call(target, matchSelector)
        ) {
            stopDetectingUsingHint();
            success();
            return true;
        }
        return false;
    }
    function setupMatchObserver(target) {
        hintMatchObserver === null || hintMatchObserver === void 0
            ? void 0
            : hintMatchObserver.disconnect();
        if (checkMatch(target)) {
            return;
        }
        hintMatchObserver = new MutationObserver(() => checkMatch(target));
        hintMatchObserver.observe(target, {attributes: true});
    }
    const target = document.querySelector(hint.target);
    if (target) {
        setupMatchObserver(target);
    } else {
        hintTargetObserver = new MutationObserver((mutations) => {
```

```
            const handledTargets = new Set();
            for (const mutation of mutations) {
                if (handledTargets.has(mutation.target)) {
                    continue;
                }
                handledTargets.add(mutation.target);
                if (mutation.target instanceof Element) {
                    const target = mutation.target.querySelector(
                        hint.target
                    );
                    if (target) {
                        hintTargetObserver.disconnect();
                        setupMatchObserver(target);
                        break;
                    }
                }
            }
        });
        hintTargetObserver.observe(document.documentElement, {
            childList: true,
            subtree: true
        });
    }
}
function stopDetectingUsingHint() {
    hintTargetObserver === null || hintTargetObserver === void 0
        ? void 0
        : hintTargetObserver.disconnect();
    hintMatchObserver === null || hintMatchObserver === void 0
        ? void 0
        : hintMatchObserver.disconnect();
}

function cachedFactory(factory, size) {
    const cache = new Map();
    return (key) => {
        if (cache.has(key)) {
            return cache.get(key);
        }
        const value = factory(key);
        cache.set(key, value);
        if (cache.size > size) {
            const first = cache.keys().next().value;
            cache.delete(first);
        }
        return value;
    };
}

const simpleIPV6Regex = /\[[0-9:a-zA-Z]+?\]/;
function isIPV6(url) {
    const openingBracketIndex = simpleIPV6Regex.exec(url);
    if (!openingBracketIndex) {
        return false;
    }
    const queryIndex = url.indexOf("?");
    if (queryIndex >= 0 && openingBracketIndex.index > queryIndex) {
        return false;
    }
    return true;
}
const ipV6HostRegex = /\[.*?\](\:\d+)?/;
function compareIPV6(firstURL, secondURL) {
    const firstHost = firstURL.match(ipV6HostRegex)[0];
    const secondHost = secondURL.match(ipV6HostRegex)[0];
    return firstHost === secondHost;
}

let anchor;
```

```
const parsedURLCache = new Map();
function fixBaseURL($url) {
    if (!anchor) {
        anchor = document.createElement("a");
    }
    anchor.href = $url;
    return anchor.href;
}
function parseURL($url, $base = null) {
    const key = `${$url}${$base ? `;${$base}` : ""}`;
    if (parsedURLCache.has(key)) {
        return parsedURLCache.get(key);
    }
    if ($base) {
        const parsedURL = new URL($url, fixBaseURL($base));
        parsedURLCache.set(key, parsedURL);
        return parsedURL;
    }
    const parsedURL = new URL(fixBaseURL($url));
    parsedURLCache.set($url, parsedURL);
    return parsedURL;
}
function getAbsoluteURL($base, $relative) {
    if ($relative.match(/^data\\?\:/)) {
        return $relative;
    }
    if (/^\/\//.test($relative)) {
        return `${location.protocol}${$relative}`;
    }
    const b = parseURL($base);
    const a = parseURL($relative, b.href);
    return a.href;
}
function isRelativeHrefOnAbsolutePath(href) {
    if (href.startsWith("data:")) {
        return true;
    }
    const url = parseURL(href);
    if (url.protocol !== location.protocol) {
        return false;
    }
    if (url.hostname !== location.hostname) {
        return false;
    }
    if (url.port !== location.port) {
        return false;
    }
    return url.pathname === location.pathname;
}
function isURLInList(url, list) {
    for (let i = 0; i < list.length; i++) {
        if (isURLMatched(url, list[i])) {
            return true;
        }
    }
    return false;
}
function isURLMatched(url, urlTemplate) {
    const isFirstIPV6 = isIPV6(url);
    const isSecondIPV6 = isIPV6(urlTemplate);
    if (isRegExp(urlTemplate)) {
        const regexp = createRegExp(urlTemplate);
        return regexp ? regexp.test(url) : false;
    } else if (isFirstIPV6 && isSecondIPV6) {
        return compareIPV6(url, urlTemplate);
    } else if (!isFirstIPV6 && !isSecondIPV6) {
        return matchURLPattern(url, urlTemplate);
    }
    return false;
```

```
}
const URL_CACHE_SIZE = 32;
const prepareURL = cachedFactory((url) => {
    let parsed;
    try {
        parsed = new URL(url);
    } catch (err) {
        return null;
    }
    const {hostname, pathname, protocol, port} = parsed;
    const hostParts = hostname.split(".").reverse();
    const pathParts = pathname.split("/").slice(1);
    if (!pathParts[pathParts.length - 1]) {
        pathParts.splice(pathParts.length - 1, 1);
    }
    return {
        hostParts,
        pathParts,
        port,
        protocol
    };
}, URL_CACHE_SIZE);
const URL_MATCH_CACHE_SIZE = 32 * 1024;
const preparePattern = cachedFactory((pattern) => {
    if (!pattern) {
        return null;
    }
    const exactStart = pattern.startsWith("^");
    const exactEnd = pattern.endsWith("$");
    if (exactStart) {
        pattern = pattern.substring(1);
    }
    if (exactEnd) {
        pattern = pattern.substring(0, pattern.length - 1);
    }
    let protocol = "";
    const protocolIndex = pattern.indexOf("://");
    if (protocolIndex > 0) {
        protocol = pattern.substring(0, protocolIndex + 1);
        pattern = pattern.substring(protocolIndex + 3);
    }
    const slashIndex = pattern.indexOf("/");
    const host =
        slashIndex < 0 ? pattern : pattern.substring(0, slashIndex);
    let hostName = host;
    let port = "*";
    const portIndex = host.indexOf(":");
    if (portIndex >= 0) {
        hostName = host.substring(0, portIndex);
        port = host.substring(portIndex + 1);
    }
    const hostParts = hostName.split(".").reverse();
    const path = slashIndex < 0 ? "" : pattern.substring(slashIndex + 1);
    const pathParts = path.split("/");
    if (!pathParts[pathParts.length - 1]) {
        pathParts.splice(pathParts.length - 1, 1);
    }
    return {
        hostParts,
        pathParts,
        port,
        exactStart,
        exactEnd,
        protocol
    };
}, URL_MATCH_CACHE_SIZE);
function matchURLPattern(url, pattern) {
    const u = prepareURL(url);
    const p = preparePattern(pattern);
```

```
    if (
        !(u && p) ||
        p.hostParts.length > u.hostParts.length ||
        (p.exactStart && p.hostParts.length !== u.hostParts.length) ||
        (p.exactEnd && p.pathParts.length !== u.pathParts.length) ||
        (p.port !== "*" && p.port !== u.port) ||
        (p.protocol && p.protocol !== u.protocol)
    ) {
        return false;
    }
    for (let i = 0; i < p.hostParts.length; i++) {
        const pHostPart = p.hostParts[i];
        const uHostPart = u.hostParts[i];
        if (pHostPart !== "*" && pHostPart !== uHostPart) {
            return false;
        }
    }
    if (
        p.hostParts.length >= 2 &&
        p.hostParts.at(-1) !== "*" &&
        (p.hostParts.length < u.hostParts.length - 1 ||
            (p.hostParts.length === u.hostParts.length - 1 &&
                u.hostParts.at(-1) !== "www"))
    ) {
        return false;
    }
    if (p.pathParts.length === 0) {
        return true;
    }
    if (p.pathParts.length > u.pathParts.length) {
        return false;
    }
    for (let i = 0; i < p.pathParts.length; i++) {
        const pPathPart = p.pathParts[i];
        const uPathPart = u.pathParts[i];
        if (pPathPart !== "*" && pPathPart !== uPathPart) {
            return false;
        }
    }
    return true;
}
function isRegExp(pattern) {
    return (
        pattern.startsWith("/") &&
        pattern.endsWith("/") &&
        pattern.length > 2
    );
}
const REGEXP_CACHE_SIZE = 1024;
const createRegExp = cachedFactory((pattern) => {
    if (pattern.startsWith("/")) {
        pattern = pattern.substring(1);
    }
    if (pattern.endsWith("/")) {
        pattern = pattern.substring(0, pattern.length - 1);
    }
    try {
        return new RegExp(pattern);
    } catch (err) {
        return null;
    }
}, REGEXP_CACHE_SIZE);

function iterateCSSRules(rules, iterate, onMediaRuleError) {
    forEach(rules, (rule) => {
        if (rule.selectorText) {
            iterate(rule);
        } else if (rule.href) {
            try {
```

```
                iterateCSSRules(
                    rule.styleSheet.cssRules,
                    iterate,
                    onMediaRuleError
                );
            } catch (err) {
                onMediaRuleError && onMediaRuleError();
            }
        } else if (rule.media) {
            const media = Array.from(rule.media);
            const isScreenOrAllOrQuery = media.some(
                (m) =>
                    m.startsWith("screen") ||
                    m.startsWith("all") ||
                    m.startsWith("(")
            );
            const isPrintOrSpeech = media.some(
                (m) => m.startsWith("print") || m.startsWith("speech")
            );
            if (isScreenOrAllOrQuery || !isPrintOrSpeech) {
                iterateCSSRules(rule.cssRules, iterate, onMediaRuleError);
            }
        } else if (rule.conditionText) {
            if (CSS.supports(rule.conditionText)) {
                iterateCSSRules(rule.cssRules, iterate, onMediaRuleError);
            }
        } else;
    });
}
const shorthandVarDependantProperties = [
    "background",
    "border",
    "border-color",
    "border-bottom",
    "border-left",
    "border-right",
    "border-top",
    "outline",
    "outline-color"
];
function iterateCSSDeclarations(style, iterate) {
    forEach(style, (property) => {
        const value = style.getPropertyValue(property).trim();
        if (!value) {
            return;
        }
        iterate(property, value);
    });
    const cssText = style.cssText;
    if (cssText.includes("var(")) {
        {
            shorthandVarDependantProperties.forEach((prop) => {
                const val = style.getPropertyValue(prop);
                if (val && val.includes("var(")) {
                    iterate(prop, val);
                }
            });
        }
    }
}
const cssURLRegex = /url\((('.*?')|(".*?")|([^\)]*?))\)/g;
const cssImportRegex =
    /@import\s*(url\()?(('.+?')|(".+?")|([^\)]*?))\)? ?(screen)?;?/gi;
function getCSSURLValue(cssURL) {
    return cssURL
        .trim()
        .replace(/[\n\r\\]+/g, "")
        .replace(/^url\((.*)\)$/, "$1")
        .trim()
```

```
        .replace(/^"(.*)"$/, "$1")
        .replace(/^'(.*)'$/, "$1")
        .replace(/(?:\\(.))/g, "$1");
}
function getCSSBaseBath(url) {
    const cssURL = parseURL(url);
    return `${cssURL.origin}${cssURL.pathname
        .replace(/\?.*$/, "")
        .replace(/(\/)([^\/]+)$/i, "$1")}`;
}
function replaceCSSRelativeURLsWithAbsolute($css, cssBasePath) {
    return $css.replace(cssURLRegex, (match) => {
        const pathValue = getCSSURLValue(match);
        try {
            return `url('${getAbsoluteURL(cssBasePath, pathValue)}')`;
        } catch (err) {
            return match;
        }
    });
}
const cssCommentsRegex = /\/\*[\s\S]*?\*\//g;
function removeCSSComments($css) {
    return $css.replace(cssCommentsRegex, "");
}
const fontFaceRegex = /@font-face\s*{[^}]*}/g;
function replaceCSSFontFace($css) {
    return $css.replace(fontFaceRegex, "");
}

function scale(x, inLow, inHigh, outLow, outHigh) {
    return ((x - inLow) * (outHigh - outLow)) / (inHigh - inLow) + outLow;
}
function clamp(x, min, max) {
    return Math.min(max, Math.max(min, x));
}
function multiplyMatrices(m1, m2) {
    const result = [];
    for (let i = 0, len = m1.length; i < len; i++) {
        result[i] = [];
        for (let j = 0, len2 = m2[0].length; j < len2; j++) {
            let sum = 0;
            for (let k = 0, len3 = m1[0].length; k < len3; k++) {
                sum += m1[i][k] * m2[k][j];
            }
            result[i][j] = sum;
        }
    }
    return result;
}

function createFilterMatrix(config) {
    let m = Matrix.identity();
    if (config.sepia !== 0) {
        m = multiplyMatrices(m, Matrix.sepia(config.sepia / 100));
    }
    if (config.grayscale !== 0) {
        m = multiplyMatrices(m, Matrix.grayscale(config.grayscale / 100));
    }
    if (config.contrast !== 100) {
        m = multiplyMatrices(m, Matrix.contrast(config.contrast / 100));
    }
    if (config.brightness !== 100) {
        m = multiplyMatrices(m, Matrix.brightness(config.brightness / 100));
    }
    if (config.mode === 1) {
        m = multiplyMatrices(m, Matrix.invertNHue());
    }
    return m;
}
```

```
function applyColorMatrix([r, g, b], matrix) {
    const rgb = [[r / 255], [g / 255], [b / 255], [1], [1]];
    const result = multiplyMatrices(matrix, rgb);
    return [0, 1, 2].map((i) =>
        clamp(Math.round(result[i][0] * 255), 0, 255)
    );
}
const Matrix = {
    identity() {
        return [
            [1, 0, 0, 0, 0],
            [0, 1, 0, 0, 0],
            [0, 0, 1, 0, 0],
            [0, 0, 0, 1, 0],
            [0, 0, 0, 0, 1]
        ];
    },
    invertNHue() {
        return [
            [0.333, -0.667, -0.667, 0, 1],
            [-0.667, 0.333, -0.667, 0, 1],
            [-0.667, -0.667, 0.333, 0, 1],
            [0, 0, 0, 1, 0],
            [0, 0, 0, 0, 1]
        ];
    },
    brightness(v) {
        return [
            [v, 0, 0, 0, 0],
            [0, v, 0, 0, 0],
            [0, 0, v, 0, 0],
            [0, 0, 0, 1, 0],
            [0, 0, 0, 0, 1]
        ];
    },
    contrast(v) {
        const t = (1 - v) / 2;
        return [
            [v, 0, 0, 0, t],
            [0, v, 0, 0, t],
            [0, 0, v, 0, t],
            [0, 0, 0, 1, 0],
            [0, 0, 0, 0, 1]
        ];
    },
    sepia(v) {
        return [
            [
                0.393 + 0.607 * (1 - v),
                0.769 - 0.769 * (1 - v),
                0.189 - 0.189 * (1 - v),
                0,
                0
            ],
            [
                0.349 - 0.349 * (1 - v),
                0.686 + 0.314 * (1 - v),
                0.168 - 0.168 * (1 - v),
                0,
                0
            ],
            [
                0.272 - 0.272 * (1 - v),
                0.534 - 0.534 * (1 - v),
                0.131 + 0.869 * (1 - v),
                0,
                0
            ],
            [0, 0, 0, 1, 0],
```

```
            [0, 0, 0, 0, 1]
        ];
    },
    grayscale(v) {
        return [
            [
                0.2126 + 0.7874 * (1 - v),
                0.7152 - 0.7152 * (1 - v),
                0.0722 - 0.0722 * (1 - v),
                0,
                0
            ],
            [
                0.2126 - 0.2126 * (1 - v),
                0.7152 + 0.2848 * (1 - v),
                0.0722 - 0.0722 * (1 - v),
                0,
                0
            ],
            [
                0.2126 - 0.2126 * (1 - v),
                0.7152 - 0.7152 * (1 - v),
                0.0722 + 0.9278 * (1 - v),
                0,
                0
            ],
            [0, 0, 0, 1, 0],
            [0, 0, 0, 0, 1]
        ];
    }
};

function getBgPole(theme) {
    const isDarkScheme = theme.mode === 1;
    const prop = isDarkScheme
        ? "darkSchemeBackgroundColor"
        : "lightSchemeBackgroundColor";
    return theme[prop];
}
function getFgPole(theme) {
    const isDarkScheme = theme.mode === 1;
    const prop = isDarkScheme
        ? "darkSchemeTextColor"
        : "lightSchemeTextColor";
    return theme[prop];
}
const colorModificationCache = new Map();
function clearColorModificationCache() {
    colorModificationCache.clear();
}
const rgbCacheKeys = ["r", "g", "b", "a"];
const themeCacheKeys$1 = [
    "mode",
    "brightness",
    "contrast",
    "grayscale",
    "sepia",
    "darkSchemeBackgroundColor",
    "darkSchemeTextColor",
    "lightSchemeBackgroundColor",
    "lightSchemeTextColor"
];
function getCacheId(rgb, theme) {
    let resultId = "";
    rgbCacheKeys.forEach((key) => {
        resultId += `${rgb[key]};`;
    });
    themeCacheKeys$1.forEach((key) => {
        resultId += `${theme[key]};`;
```

```
    });
    return resultId;
}
function modifyColorWithCache(
    rgb,
    theme,
    modifyHSL,
    poleColor,
    anotherPoleColor
) {
    let fnCache;
    if (colorModificationCache.has(modifyHSL)) {
        fnCache = colorModificationCache.get(modifyHSL);
    } else {
        fnCache = new Map();
        colorModificationCache.set(modifyHSL, fnCache);
    }
    const id = getCacheId(rgb, theme);
    if (fnCache.has(id)) {
        return fnCache.get(id);
    }
    const hsl = rgbToHSL(rgb);
    const pole = poleColor == null ? null : parseToHSLWithCache(poleColor);
    const anotherPole =
        anotherPoleColor == null
            ? null
            : parseToHSLWithCache(anotherPoleColor);
    const modified = modifyHSL(hsl, pole, anotherPole);
    const {r, g, b, a} = hslToRGB(modified);
    const matrix = createFilterMatrix(theme);
    const [rf, gf, bf] = applyColorMatrix([r, g, b], matrix);
    const color =
        a === 1
            ? rgbToHexString({r: rf, g: gf, b: bf})
            : rgbToString({r: rf, g: gf, b: bf, a});
    fnCache.set(id, color);
    return color;
}
function noopHSL(hsl) {
    return hsl;
}
function modifyColor(rgb, theme) {
    return modifyColorWithCache(rgb, theme, noopHSL);
}
function modifyLightSchemeColor(rgb, theme) {
    const poleBg = getBgPole(theme);
    const poleFg = getFgPole(theme);
    return modifyColorWithCache(
        rgb,
        theme,
        modifyLightModeHSL,
        poleFg,
        poleBg
    );
}
function modifyLightModeHSL({h, s, l, a}, poleFg, poleBg) {
    const isDark = l < 0.5;
    let isNeutral;
    if (isDark) {
        isNeutral = l < 0.2 || s < 0.12;
    } else {
        const isBlue = h > 200 && h < 280;
        isNeutral = s < 0.24 || (l > 0.8 && isBlue);
    }
    let hx = h;
    let sx = l;
    if (isNeutral) {
        if (isDark) {
            hx = poleFg.h;
```

```
            sx = poleFg.s;
        } else {
            hx = poleBg.h;
            sx = poleBg.s;
        }
    }
    const lx = scale(l, 0, 1, poleFg.l, poleBg.l);
    return {h: hx, s: sx, l: lx, a};
}
const MAX_BG_LIGHTNESS = 0.4;
function modifyBgHSL({h, s, l, a}, pole) {
    const isDark = l < 0.5;
    const isBlue = h > 200 && h < 280;
    const isNeutral = s < 0.12 || (l > 0.8 && isBlue);
    if (isDark) {
        const lx = scale(l, 0, 0.5, 0, MAX_BG_LIGHTNESS);
        if (isNeutral) {
            const hx = pole.h;
            const sx = pole.s;
            return {h: hx, s: sx, l: lx, a};
        }
        return {h, s, l: lx, a};
    }
    let lx = scale(l, 0.5, 1, MAX_BG_LIGHTNESS, pole.l);
    if (isNeutral) {
        const hx = pole.h;
        const sx = pole.s;
        return {h: hx, s: sx, l: lx, a};
    }
    let hx = h;
    const isYellow = h > 60 && h < 180;
    if (isYellow) {
        const isCloserToGreen = h > 120;
        if (isCloserToGreen) {
            hx = scale(h, 120, 180, 135, 180);
        } else {
            hx = scale(h, 60, 120, 60, 105);
        }
    }
    if (hx > 40 && hx < 80) {
        lx *= 0.75;
    }
    return {h: hx, s, l: lx, a};
}
function modifyBackgroundColor(rgb, theme) {
    if (theme.mode === 0) {
        return modifyLightSchemeColor(rgb, theme);
    }
    const pole = getBgPole(theme);
    return modifyColorWithCache(
        rgb,
        {...theme, mode: 0},
        modifyBgHSL,
        pole
    );
}
const MIN_FG_LIGHTNESS = 0.55;
function modifyBlueFgHue(hue) {
    return scale(hue, 205, 245, 205, 220);
}
function modifyFgHSL({h, s, l, a}, pole) {
    const isLight = l > 0.5;
    const isNeutral = l < 0.2 || s < 0.24;
    const isBlue = !isNeutral && h > 205 && h < 245;
    if (isLight) {
        const lx = scale(l, 0.5, 1, MIN_FG_LIGHTNESS, pole.l);
        if (isNeutral) {
            const hx = pole.h;
            const sx = pole.s;
```

```
            return {h: hx, s: sx, l: lx, a};
        }
        let hx = h;
        if (isBlue) {
            hx = modifyBlueFgHue(h);
        }
        return {h: hx, s, l: lx, a};
    }
    if (isNeutral) {
        const hx = pole.h;
        const sx = pole.s;
        const lx = scale(l, 0, 0.5, pole.l, MIN_FG_LIGHTNESS);
        return {h: hx, s: sx, l: lx, a};
    }
    let hx = h;
    let lx;
    if (isBlue) {
        hx = modifyBlueFgHue(h);
        lx = scale(l, 0, 0.5, pole.l, Math.min(1, MIN_FG_LIGHTNESS + 0.05));
    } else {
        lx = scale(l, 0, 0.5, pole.l, MIN_FG_LIGHTNESS);
    }
    return {h: hx, s, l: lx, a};
}
function modifyForegroundColor(rgb, theme) {
    if (theme.mode === 0) {
        return modifyLightSchemeColor(rgb, theme);
    }
    const pole = getFgPole(theme);
    return modifyColorWithCache(
        rgb,
        {...theme, mode: 0},
        modifyFgHSL,
        pole
    );
}
function modifyBorderHSL({h, s, l, a}, poleFg, poleBg) {
    const isDark = l < 0.5;
    const isNeutral = l < 0.2 || s < 0.24;
    let hx = h;
    let sx = s;
    if (isNeutral) {
        if (isDark) {
            hx = poleFg.h;
            sx = poleFg.s;
        } else {
            hx = poleBg.h;
            sx = poleBg.s;
        }
    }
    const lx = scale(l, 0, 1, 0.5, 0.2);
    return {h: hx, s: sx, l: lx, a};
}
function modifyBorderColor(rgb, theme) {
    if (theme.mode === 0) {
        return modifyLightSchemeColor(rgb, theme);
    }
    const poleFg = getFgPole(theme);
    const poleBg = getBgPole(theme);
    return modifyColorWithCache(
        rgb,
        {...theme, mode: 0},
        modifyBorderHSL,
        poleFg,
        poleBg
    );
}
function modifyShadowColor(rgb, filter) {
    return modifyBackgroundColor(rgb, filter);
```

```
    }
    function modifyGradientColor(rgb, filter) {
        return modifyBackgroundColor(rgb, filter);
    }

    function createTextStyle(config) {
        const lines = [];
        lines.push(
            '*:not(pre, pre *, code, .far, .fa, .glyphicon, [class*="vjs-"], .fab, .fa-github, .fas,
.material-icons, .icofont, .typcn, mu, [class*="mu-"], .glyphicon, .icon) {'
        );
        if (config.useFont && config.fontFamily) {
            lines.push(`  font-family: ${config.fontFamily} !important;`);
        }
        if (config.textStroke > 0) {
            lines.push(
                `  -webkit-text-stroke: ${config.textStroke}px !important;`
            );
            lines.push(`  text-stroke: ${config.textStroke}px !important;`);
        }
        lines.push("}");
        return lines.join("\n");
    }

    var FilterMode;
    (function (FilterMode) {
        FilterMode[(FilterMode["light"] = 0)] = "light";
        FilterMode[(FilterMode["dark"] = 1)] = "dark";
    })(FilterMode || (FilterMode = {}));
    function getCSSFilterValue(config) {
        const filters = [];
        if (config.mode === FilterMode.dark) {
            filters.push("invert(100%) hue-rotate(180deg)");
        }
        if (config.brightness !== 100) {
            filters.push(`brightness(${config.brightness}%)`);
        }
        if (config.contrast !== 100) {
            filters.push(`contrast(${config.contrast}%)`);
        }
        if (config.grayscale !== 0) {
            filters.push(`grayscale(${config.grayscale}%)`);
        }
        if (config.sepia !== 0) {
            filters.push(`sepia(${config.sepia}%)`);
        }
        if (filters.length === 0) {
            return null;
        }
        return filters.join(" ");
    }

    function toSVGMatrix(matrix) {
        return matrix
            .slice(0, 4)
            .map((m) => m.map((m) => m.toFixed(3)).join(" "))
            .join(" ");
    }
    function getSVGFilterMatrixValue(config) {
        return toSVGMatrix(createFilterMatrix(config));
    }

    function hexify(number) {
        return (number < 16 ? "0" : "") + number.toString(16);
    }
    function generateUID() {
        if ("randomUUID" in crypto) {
            const uuid = crypto.randomUUID();
            return (
```

```
            uuid.substring(0, 8) +
            uuid.substring(9, 13) +
            uuid.substring(14, 18) +
            uuid.substring(19, 23) +
            uuid.substring(24)
        );
    }
    if ("getRandomValues" in crypto) {
        return Array.from(crypto.getRandomValues(new Uint8Array(16)))
            .map((x) => hexify(x))
            .join("");
    }
    return Math.floor(Math.random() * 2 ** 55).toString(36);
}

const resolvers$1 = new Map();
const rejectors = new Map();
async function bgFetch(request) {
    return new Promise((resolve, reject) => {
        const id = generateUID();
        resolvers$1.set(id, resolve);
        rejectors.set(id, reject);
        chrome.runtime.sendMessage({
            type: MessageTypeCStoBG.FETCH,
            data: request,
            id
        });
    });
}
chrome.runtime.onMessage.addListener(({type, data, error, id}) => {
    if (type === MessageTypeBGtoCS.FETCH_RESPONSE) {
        const resolve = resolvers$1.get(id);
        const reject = rejectors.get(id);
        resolvers$1.delete(id);
        rejectors.delete(id);
        if (error) {
            reject && reject(error);
        } else {
            resolve && resolve(data);
        }
    }
});

async function getOKResponse(url, mimeType, origin) {
    const response = await fetch(url, {
        cache: "force-cache",
        credentials: "omit",
        referrer: origin
    });
    if (
        mimeType &&
        !response.headers.get("Content-Type").startsWith(mimeType)
    ) {
        throw new Error(`Mime type mismatch when loading ${url}`);
    }
    if (!response.ok) {
        throw new Error(
            `Unable to load ${url} ${response.status} ${response.statusText}`
        );
    }
    return response;
}
async function loadAsDataURL(url, mimeType) {
    const response = await getOKResponse(url, mimeType);
    return await readResponseAsDataURL(response);
}
async function loadAsBlob(url, mimeType) {
    const response = await getOKResponse(url, mimeType);
    return await response.blob();
```

```
}
async function readResponseAsDataURL(response) {
    const blob = await response.blob();
    const dataURL = await new Promise((resolve) => {
        const reader = new FileReader();
        reader.onloadend = () => resolve(reader.result);
        reader.readAsDataURL(blob);
    });
    return dataURL;
}

const MAX_FRAME_DURATION = 1000 / 60;
class AsyncQueue {
    constructor() {
        this.queue = [];
        this.timerId = null;
    }
    addTask(task) {
        this.queue.push(task);
        this.scheduleFrame();
    }
    stop() {
        if (this.timerId !== null) {
            cancelAnimationFrame(this.timerId);
            this.timerId = null;
        }
        this.queue = [];
    }
    scheduleFrame() {
        if (this.timerId) {
            return;
        }
        this.timerId = requestAnimationFrame(() => {
            this.timerId = null;
            const start = Date.now();
            let cb;
            while ((cb = this.queue.shift())) {
                cb();
                if (Date.now() - start >= MAX_FRAME_DURATION) {
                    this.scheduleFrame();
                    break;
                }
            }
        });
    }
}

const imageManager = new AsyncQueue();
async function getImageDetails(url) {
    return new Promise(async (resolve, reject) => {
        var _a, _b;
        try {
            const dataURL = url.startsWith("data:")
                ? url
                : await getDataURL(url);
            const blob =
                (_a = tryConvertDataURLToBlobSync(dataURL)) !== null &&
                _a !== void 0
                    ? _a
                    : await loadAsBlob(url);
            let image;
            if (dataURL.startsWith("data:image/svg+xml")) {
                image = await loadImage(dataURL);
            } else {
                image =
                    (_b = await tryCreateImageBitmap(blob)) !== null &&
                    _b !== void 0
                        ? _b
                        : await loadImage(dataURL);
```

```
            }
            imageManager.addTask(() => {
                const analysis = analyzeImage(image);
                resolve({
                    src: url,
                    blob,
                    dataURL,
                    width: image.width,
                    height: image.height,
                    ...analysis
                });
            });
        } catch (error) {
            reject(error);
        }
    });
}
async function getDataURL(url) {
    const parsedURL = new URL(url);
    if (parsedURL.origin === location.origin) {
        return await loadAsDataURL(url);
    }
    return await bgFetch({url, responseType: "data-url"});
}
async function tryCreateImageBitmap(blob) {
    try {
        return await createImageBitmap(blob);
    } catch (err) {
        logWarn(
            `Unable to create image bitmap for type ${blob.type}: ${String(
                err
            )}`
        );
        return null;
    }
}
const INCOMPLETE_DOC_LOADING_IMAGE_LIMIT = 256;
let loadingImagesCount = 0;
async function loadImage(url) {
    return new Promise((resolve, reject) => {
        const image = new Image();
        image.onload = () => resolve(image);
        image.onerror = () => reject(`Unable to load image ${url}`);
        if (
            ++loadingImagesCount <= INCOMPLETE_DOC_LOADING_IMAGE_LIMIT ||
            isReadyStateComplete()
        ) {
            image.src = url;
        } else {
            addReadyStateCompleteListener(() => (image.src = url));
        }
    });
}
const MAX_ANALYSIS_PIXELS_COUNT = 32 * 32;
let canvas;
let context;
function createCanvas() {
    const maxWidth = MAX_ANALYSIS_PIXELS_COUNT;
    const maxHeight = MAX_ANALYSIS_PIXELS_COUNT;
    canvas = document.createElement("canvas");
    canvas.width = maxWidth;
    canvas.height = maxHeight;
    context = canvas.getContext("2d", {willReadFrequently: true});
    context.imageSmoothingEnabled = false;
}
function removeCanvas() {
    canvas = null;
    context = null;
}
```

```
const LARGE_IMAGE_PIXELS_COUNT = 512 * 512;
function analyzeImage(image) {
    if (!canvas) {
        createCanvas();
    }
    let sw;
    let sh;
    if (image instanceof HTMLImageElement) {
        sw = image.naturalWidth;
        sh = image.naturalHeight;
    } else {
        sw = image.width;
        sh = image.height;
    }
    if (sw === 0 || sh === 0) {
        return {
            isDark: false,
            isLight: false,
            isTransparent: false,
            isLarge: false,
            isTooLarge: false
        };
    }
    if (sw * sh > LARGE_IMAGE_PIXELS_COUNT) {
        return {
            isDark: false,
            isLight: false,
            isTransparent: false,
            isLarge: true
        };
    }
    const sourcePixelsCount = sw * sh;
    const k = Math.min(
        1,
        Math.sqrt(MAX_ANALYSIS_PIXELS_COUNT / sourcePixelsCount)
    );
    const width = Math.ceil(sw * k);
    const height = Math.ceil(sh * k);
    context.clearRect(0, 0, width, height);
    context.drawImage(image, 0, 0, sw, sh, 0, 0, width, height);
    const imageData = context.getImageData(0, 0, width, height);
    const d = imageData.data;
    const TRANSPARENT_ALPHA_THRESHOLD = 0.05;
    const DARK_LIGHTNESS_THRESHOLD = 0.4;
    const LIGHT_LIGHTNESS_THRESHOLD = 0.7;
    let transparentPixelsCount = 0;
    let darkPixelsCount = 0;
    let lightPixelsCount = 0;
    let i, x, y;
    let r, g, b, a;
    let l;
    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            i = 4 * (y * width + x);
            r = d[i + 0];
            g = d[i + 1];
            b = d[i + 2];
            a = d[i + 3];
            if (a / 255 < TRANSPARENT_ALPHA_THRESHOLD) {
                transparentPixelsCount++;
            } else {
                l = getSRGBLightness(r, g, b);
                if (l < DARK_LIGHTNESS_THRESHOLD) {
                    darkPixelsCount++;
                }
                if (l > LIGHT_LIGHTNESS_THRESHOLD) {
                    lightPixelsCount++;
                }
            }
```

```
        }
    }
    const totalPixelsCount = width * height;
    const opaquePixelsCount = totalPixelsCount - transparentPixelsCount;
    const DARK_IMAGE_THRESHOLD = 0.7;
    const LIGHT_IMAGE_THRESHOLD = 0.7;
    const TRANSPARENT_IMAGE_THRESHOLD = 0.1;
    return {
        isDark: darkPixelsCount / opaquePixelsCount >= DARK_IMAGE_THRESHOLD,
        isLight:
            lightPixelsCount / opaquePixelsCount >= LIGHT_IMAGE_THRESHOLD,
        isTransparent:
            transparentPixelsCount / totalPixelsCount >=
            TRANSPARENT_IMAGE_THRESHOLD,
        isLarge: false
    };
}
let isBlobURLSupported = null;
let canUseProxy = false;
let blobURLCheckRequested = false;
const blobURLCheckAwaiters = [];
document.addEventListener(
    "__darkreader__inlineScriptsAllowed",
    () => (canUseProxy = true),
    {once: true}
);
async function requestBlobURLCheck() {
    if (!canUseProxy) {
        return;
    }
    if (blobURLCheckRequested) {
        return await new Promise((resolve) =>
            blobURLCheckAwaiters.push(resolve)
        );
    }
    blobURLCheckRequested = true;
    await new Promise((resolve) => {
        document.addEventListener(
            "__darkreader__blobURLCheckResponse",
            (e) => {
                isBlobURLSupported = e.detail.blobURLAllowed;
                resolve();
                blobURLCheckAwaiters.forEach((r) => r());
                blobURLCheckAwaiters.splice(0);
            },
            {once: true}
        );
        document.dispatchEvent(
            new CustomEvent("__darkreader__blobURLCheckRequest")
        );
    });
}
function isBlobURLCheckResultReady() {
    return isBlobURLSupported != null || !canUseProxy;
}
function onCSPError(err) {
    if (err.blockedURI === "blob") {
        isBlobURLSupported = false;
        document.removeEventListener("securitypolicyviolation", onCSPError);
    }
}
document.addEventListener("securitypolicyviolation", onCSPError);
const objectURLs = new Set();
function getFilteredImageURL({dataURL, width, height}, theme) {
    if (dataURL.startsWith("data:image/svg+xml")) {
        dataURL = escapeXML(dataURL);
    }
    const matrix = getSVGFilterMatrixValue(theme);
    const svg = [
```

```
            `<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" width="${width}"
height="${height}">`,
            "<defs>",
            '<filter id="darkreader-image-filter">',
            `<feColorMatrix type="matrix" values="${matrix}" />`,
            "</filter>",
            "</defs>",
            `<image width="${width}" height="${height}" filter="url(#darkreader-image-filter)"
xlink:href="${dataURL}" />`,
            "</svg>"
        ].join("");
        if (!isBlobURLSupported) {
            return `${btoa(svg)}`;
        }
        const bytes = new Uint8Array(svg.length);
        for (let i = 0; i < svg.length; i++) {
            bytes[i] = svg.charCodeAt(i);
        }
        const blob = new Blob([bytes], {type: "image/svg+xml"});
        const objectURL = URL.createObjectURL(blob);
        objectURLs.add(objectURL);
        return objectURL;
    }
    const xmlEscapeChars = {
        "<": "&lt;",
        ">": "&gt;",
        "&": "&amp;",
        "'": "&apos;",
        '"': "&quot;"
    };
    function escapeXML(str) {
        return str.replace(/[<>&'"]/g, (c) => {
            var _a;
            return (_a = xmlEscapeChars[c]) !== null && _a !== void 0 ? _a : c;
        });
    }
    const dataURLBlobURLs = new Map();
    function tryConvertDataURLToBlobSync(dataURL) {
        const colonIndex = dataURL.indexOf(":");
        const semicolonIndex = dataURL.indexOf(";", colonIndex + 1);
        const commaIndex = dataURL.indexOf(",", semicolonIndex + 1);
        const encoding = dataURL
            .substring(semicolonIndex + 1, commaIndex)
            .toLocaleLowerCase();
        const mediaType = dataURL.substring(colonIndex + 1, semicolonIndex);
        if (encoding !== "base64" || !mediaType) {
            return null;
        }
        const characters = atob(dataURL.substring(commaIndex + 1));
        const bytes = new Uint8Array(characters.length);
        for (let i = 0; i < characters.length; i++) {
            bytes[i] = characters.charCodeAt(i);
        }
        return new Blob([bytes], {type: mediaType});
    }
    async function tryConvertDataURLToBlobURL(dataURL) {
        if (!isBlobURLSupported) {
            return null;
        }
        let blobURL = dataURLBlobURLs.get(dataURL);
        if (blobURL) {
            return blobURL;
        }
        let blob = tryConvertDataURLToBlobSync(dataURL);
        if (!blob) {
            const response = await fetch(dataURL);
            blob = await response.blob();
        }
        blobURL = URL.createObjectURL(blob);
```

```
    dataURLBlobURLs.set(dataURL, blobURL);
    return blobURL;
}
function cleanImageProcessingCache() {
    imageManager && imageManager.stop();
    removeCanvas();
    objectURLs.forEach((u) => URL.revokeObjectURL(u));
    objectURLs.clear();
    dataURLBlobURLs.forEach((u) => URL.revokeObjectURL(u));
    dataURLBlobURLs.clear();
}

const gradientLength = "gradient".length;
const conicGradient = "conic-";
const conicGradientLength = conicGradient.length;
const radialGradient = "radial-";
const linearGradient = "linear-";
function parseGradient(value) {
    const result = [];
    let index = 0;
    let startIndex = conicGradient.length;
    while ((index = value.indexOf("gradient", startIndex)) !== -1) {
        let typeGradient;
        [linearGradient, radialGradient, conicGradient].find(
            (possibleType) => {
                if (index - possibleType.length >= 0) {
                    const possibleGradient = value.substring(
                        index - possibleType.length,
                        index
                    );
                    if (possibleGradient === possibleType) {
                        if (
                            value.slice(
                                index - possibleType.length - 10,
                                index - possibleType.length - 1
                            ) === "repeating"
                        ) {
                            typeGradient = `repeating-${possibleType}gradient`;
                            return true;
                        }
                        if (
                            value.slice(
                                index - possibleType.length - 8,
                                index - possibleType.length - 1
                            ) === "-webkit"
                        ) {
                            typeGradient = `-webkit-${possibleType}gradient`;
                            return true;
                        }
                        typeGradient = `${possibleType}gradient`;
                        return true;
                    }
                }
            }
        );
        if (!typeGradient) {
            break;
        }
        const {start, end} = getParenthesesRange(
            value,
            index + gradientLength
        );
        const match = value.substring(start + 1, end - 1);
        startIndex = end + 1 + conicGradientLength;
        result.push({
            typeGradient,
            match,
            offset: typeGradient.length + 2,
            index: index - typeGradient.length + gradientLength,
```

```
            hasComma: true
        });
    }
    if (result.length) {
        result[result.length - 1].hasComma = false;
    }
    return result;
}

function getPriority(ruleStyle, property) {
    return Boolean(ruleStyle && ruleStyle.getPropertyPriority(property));
}
function getModifiableCSSDeclaration(
    property,
    value,
    rule,
    variablesStore,
    ignoreImageSelectors,
    isCancelled
) {
    if (property.startsWith("--")) {
        const modifier = getVariableModifier(
            variablesStore,
            property,
            value,
            rule,
            ignoreImageSelectors,
            isCancelled
        );
        if (modifier) {
            return {
                property,
                value: modifier,
                important: getPriority(rule.style, property),
                sourceValue: value
            };
        }
    } else if (value.includes("var(")) {
        const modifier = getVariableDependantModifier(
            variablesStore,
            property,
            value
        );
        if (modifier) {
            return {
                property,
                value: modifier,
                important: getPriority(rule.style, property),
                sourceValue: value
            };
        }
    } else if (property === "color-scheme") {
        return null;
    } else if (
        (property.includes("color") &&
            property !== "-webkit-print-color-adjust") ||
        property === "fill" ||
        property === "stroke" ||
        property === "stop-color"
    ) {
        const modifier = getColorModifier(property, value, rule);
        if (modifier) {
            return {
                property,
                value: modifier,
                important: getPriority(rule.style, property),
                sourceValue: value
            };
        }
```

```
    } else if (
        property === "background-image" ||
        property === "list-style-image"
    ) {
        const modifier = getBgImageModifier(
            value,
            rule,
            ignoreImageSelectors,
            isCancelled
        );
        if (modifier) {
            return {
                property,
                value: modifier,
                important: getPriority(rule.style, property),
                sourceValue: value
            };
        }
    } else if (property.includes("shadow")) {
        const modifier = getShadowModifier(value);
        if (modifier) {
            return {
                property,
                value: modifier,
                important: getPriority(rule.style, property),
                sourceValue: value
            };
        }
    }
    return null;
}
function joinSelectors(...selectors) {
    return selectors.filter(Boolean).join(", ");
}
function getModifiedUserAgentStyle(theme, isIFrame, styleSystemControls) {
    const lines = [];
    if (!isIFrame) {
        lines.push("html {");
        lines.push(
            `    background-color: ${modifyBackgroundColor(
                {r: 255, g: 255, b: 255},
                theme
            )} !important;`
        );
        lines.push("}");
    }
    if (isCSSColorSchemePropSupported) {
        lines.push("html {");
        lines.push(
            `    color-scheme: ${
                theme.mode === 1 ? "dark" : "dark light"
            } !important;`
        );
        lines.push("}");
    }
    const bgSelectors = joinSelectors(
        isIFrame ? "" : "html, body",
        styleSystemControls ? "input, textarea, select, button, dialog" : ""
    );
    if (bgSelectors) {
        lines.push(`${bgSelectors} {`);
        lines.push(
            `    background-color: ${modifyBackgroundColor(
                {r: 255, g: 255, b: 255},
                theme
            )};`
        );
        lines.push("}");
    }
```

```
    lines.push(
        `${joinSelectors(
            "html, body",
            styleSystemControls ? "input, textarea, select, button" : ""
        )} {`
    );
    lines.push(
        `    border-color: ${modifyBorderColor(
            {r: 76, g: 76, b: 76},
            theme
        )};`
    );
    lines.push(
        `    color: ${modifyForegroundColor({r: 0, g: 0, b: 0}, theme)};`
    );
    lines.push("}");
    lines.push("a {");
    lines.push(
        `    color: ${modifyForegroundColor({r: 0, g: 64, b: 255}, theme)};`
    );
    lines.push("}");
    lines.push("table {");
    lines.push(
        `    border-color: ${modifyBorderColor(
            {r: 128, g: 128, b: 128},
            theme
        )};`
    );
    lines.push("}");
    lines.push("::placeholder {");
    lines.push(
        `    color: ${modifyForegroundColor(
            {r: 169, g: 169, b: 169},
            theme
        )};`
    );
    lines.push("}");
    lines.push("input:-webkit-autofill,");
    lines.push("textarea:-webkit-autofill,");
    lines.push("select:-webkit-autofill {");
    lines.push(
        `    background-color: ${modifyBackgroundColor(
            {r: 250, g: 255, b: 189},
            theme
        )} !important;`
    );
    lines.push(
        `    color: ${modifyForegroundColor(
            {r: 0, g: 0, b: 0},
            theme
        )} !important;`
    );
    lines.push("}");
    if (theme.scrollbarColor) {
        lines.push(getModifiedScrollbarStyle(theme));
    }
    if (theme.selectionColor) {
        lines.push(getModifiedSelectionStyle(theme));
    }
    return lines.join("\n");
}
function getSelectionColor(theme) {
    let backgroundColorSelection;
    let foregroundColorSelection;
    if (theme.selectionColor === "auto") {
        backgroundColorSelection = modifyBackgroundColor(
            {r: 0, g: 96, b: 212},
            {...theme, grayscale: 0}
        );
```

```
        foregroundColorSelection = modifyForegroundColor(
            {r: 255, g: 255, b: 255},
            {...theme, grayscale: 0}
        );
    } else {
        const rgb = parseColorWithCache(theme.selectionColor);
        const hsl = rgbToHSL(rgb);
        backgroundColorSelection = theme.selectionColor;
        if (hsl.l < 0.5) {
            foregroundColorSelection = "#FFF";
        } else {
            foregroundColorSelection = "#000";
        }
    }
    return {backgroundColorSelection, foregroundColorSelection};
}
function getModifiedSelectionStyle(theme) {
    const lines = [];
    const modifiedSelectionColor = getSelectionColor(theme);
    const backgroundColorSelection =
        modifiedSelectionColor.backgroundColorSelection;
    const foregroundColorSelection =
        modifiedSelectionColor.foregroundColorSelection;
    ["::selection", "::-moz-selection"].forEach((selection) => {
        lines.push(`${selection} {`);
        lines.push(
            `    background-color: ${backgroundColorSelection} !important;`
        );
        lines.push(`    color: ${foregroundColorSelection} !important;`);
        lines.push("}");
    });
    return lines.join("\n");
}
function getModifiedScrollbarStyle(theme) {
    const lines = [];
    let colorTrack;
    let colorIcons;
    let colorThumb;
    let colorThumbHover;
    let colorThumbActive;
    let colorCorner;
    if (theme.scrollbarColor === "auto") {
        colorTrack = modifyBackgroundColor({r: 241, g: 241, b: 241}, theme);
        colorIcons = modifyForegroundColor({r: 96, g: 96, b: 96}, theme);
        colorThumb = modifyBackgroundColor({r: 176, g: 176, b: 176}, theme);
        colorThumbHover = modifyBackgroundColor(
            {r: 144, g: 144, b: 144},
            theme
        );
        colorThumbActive = modifyBackgroundColor(
            {r: 96, g: 96, b: 96},
            theme
        );
        colorCorner = modifyBackgroundColor(
            {r: 255, g: 255, b: 255},
            theme
        );
    } else {
        const rgb = parseColorWithCache(theme.scrollbarColor);
        const hsl = rgbToHSL(rgb);
        const isLight = hsl.l > 0.5;
        const lighten = (lighter) => ({
            ...hsl,
            l: clamp(hsl.l + lighter, 0, 1)
        });
        const darken = (darker) => ({
            ...hsl,
            l: clamp(hsl.l - darker, 0, 1)
        });
```

```
        colorTrack = hslToString(darken(0.4));
        colorIcons = hslToString(isLight ? darken(0.4) : lighten(0.4));
        colorThumb = hslToString(hsl);
        colorThumbHover = hslToString(lighten(0.1));
        colorThumbActive = hslToString(lighten(0.2));
        colorCorner = hslToString(darken(0.5));
    }
    lines.push("::-webkit-scrollbar {");
    lines.push(`    background-color: ${colorTrack};`);
    lines.push(`    color: ${colorIcons};`);
    lines.push("}");
    lines.push("::-webkit-scrollbar-thumb {");
    lines.push(`    background-color: ${colorThumb};`);
    lines.push("}");
    lines.push("::-webkit-scrollbar-thumb:hover {");
    lines.push(`    background-color: ${colorThumbHover};`);
    lines.push("}");
    lines.push("::-webkit-scrollbar-thumb:active {");
    lines.push(`    background-color: ${colorThumbActive};`);
    lines.push("}");
    lines.push("::-webkit-scrollbar-corner {");
    lines.push(`    background-color: ${colorCorner};`);
    lines.push("}");
    return lines.join("\n");
}
function getModifiedFallbackStyle(filter, {strict}) {
    const factory = defaultFallbackFactory;
    return factory(filter, {strict});
}
function defaultFallbackFactory(filter, {strict}) {
    const lines = [];
    const isMicrosoft = ["microsoft.com", "docs.microsoft.com"].includes(
        location.hostname
    );
    lines.push(
        `html, body, ${
            strict
                ? `body :not(iframe)${
                      isMicrosoft
                          ? ':not(div[style^="position:absolute;top:0;left:-"]'
                          : ""
                  }`
                : "body > :not(iframe)"
        } {`
    );
    lines.push(
        `    background-color: ${modifyBackgroundColor(
            {r: 255, g: 255, b: 255},
            filter
        )} !important;`
    );
    lines.push(
        `    border-color: ${modifyBorderColor(
            {r: 64, g: 64, b: 64},
            filter
        )} !important;`
    );
    lines.push(
        `    color: ${modifyForegroundColor(
            {r: 0, g: 0, b: 0},
            filter
        )} !important;`
    );
    lines.push("}");
    return lines.join("\n");
}
const unparsableColors = new Set([
    "inherit",
    "transparent",
```

```
    "initial",
    "currentcolor",
    "none",
    "unset"
]);
function getColorModifier(prop, value, rule) {
    if (unparsableColors.has(value.toLowerCase())) {
        return value;
    }
    const rgb = parseColorWithCache(value);
    if (!rgb) {
        return null;
    }
    if (prop.includes("background")) {
        if (
            (rule.style.webkitMaskImage &&
                rule.style.webkitMaskImage !== "none") ||
            (rule.style.webkitMask &&
                !rule.style.webkitMask.startsWith("none")) ||
            (rule.style.mask && rule.style.mask !== "none") ||
            (rule.style.getPropertyValue("mask-image") &&
                rule.style.getPropertyValue("mask-image") !== "none")
        ) {
            return (filter) => modifyForegroundColor(rgb, filter);
        }
        return (filter) => modifyBackgroundColor(rgb, filter);
    }
    if (prop.includes("border") || prop.includes("outline")) {
        return (filter) => modifyBorderColor(rgb, filter);
    }
    return (filter) => modifyForegroundColor(rgb, filter);
}
const imageDetailsCache = new Map();
const awaitingForImageLoading = new Map();
function shouldIgnoreImage(selectorText, selectors) {
    if (!selectorText || selectors.length === 0) {
        return false;
    }
    if (selectors.some((s) => s === "*")) {
        return true;
    }
    const ruleSelectors = selectorText.split(/,\s*/g);
    for (let i = 0; i < selectors.length; i++) {
        const ignoredSelector = selectors[i];
        if (ruleSelectors.some((s) => s === ignoredSelector)) {
            return true;
        }
    }
    return false;
}
function getBgImageModifier(
    value,
    rule,
    ignoreImageSelectors,
    isCancelled
) {
    try {
        const gradients = parseGradient(value);
        const urls = getMatches(cssURLRegex, value);
        if (urls.length === 0 && gradients.length === 0) {
            return value;
        }
        const getIndices = (matches) => {
            let index = 0;
            return matches.map((match) => {
                const valueIndex = value.indexOf(match, index);
                index = valueIndex + match.length;
                return {match, index: valueIndex};
            });
```

```
};
const matches = gradients
    .map((i) => ({type: "gradient", ...i}))
    .concat(
        getIndices(urls).map((i) => ({
            type: "url",
            offset: 0,
            ...i
        }))
    )
    .sort((a, b) => (a.index > b.index ? 1 : -1));
const getGradientModifier = (gradient) => {
    const {typeGradient, match, hasComma} = gradient;
    const partsRegex =
        /([^\(\),]+(\([^\(\)]*(\([^\(\)]*\)*[^\(\)]*)?\))?([^\(\), ]|( (?!calc)))*),?/g;
    const colorStopRegex =
        /^(from|color-stop|to)\(([^\(\)]*?,\s*)?(.*?)\)$/;
    const parts = getMatches(partsRegex, match, 1).map((part) => {
        part = part.trim();
        let rgb = parseColorWithCache(part);
        if (rgb) {
            return (filter) => modifyGradientColor(rgb, filter);
        }
        const space = part.lastIndexOf(" ");
        rgb = parseColorWithCache(part.substring(0, space));
        if (rgb) {
            return (filter) =>
                `${modifyGradientColor(
                    rgb,
                    filter
                )} ${part.substring(space + 1)}`;
        }
        const colorStopMatch = part.match(colorStopRegex);
        if (colorStopMatch) {
            rgb = parseColorWithCache(colorStopMatch[3]);
            if (rgb) {
                return (filter) =>
                    `${colorStopMatch[1]}(${
                        colorStopMatch[2]
                            ? `${colorStopMatch[2]}, `
                            : ""
                    }${modifyGradientColor(rgb, filter)})`;
            }
        }
        return () => part;
    });
    return (filter) => {
        return `${typeGradient}(${parts
            .map((modify) => modify(filter))
            .join(", ")})${hasComma ? ", " : ""}`;
    };
};
const getURLModifier = (urlValue) => {
    var _a;
    if (
        shouldIgnoreImage(rule.selectorText, ignoreImageSelectors)
    ) {
        return null;
    }
    let url = getCSSURLValue(urlValue);
    const isURLEmpty = url.length === 0;
    const {parentStyleSheet} = rule;
    const baseURL =
        parentStyleSheet && parentStyleSheet.href
            ? getCSSBaseBath(parentStyleSheet.href)
            : ((_a = parentStyleSheet.ownerNode) === null ||
              _a === void 0
                  ? void 0
                  : _a.baseURI) || location.origin;
```

```
    url = getAbsoluteURL(baseURL, url);
    return async (filter) => {
        if (isURLEmpty) {
            return "url('')";
        }
        let imageDetails = null;
        if (imageDetailsCache.has(url)) {
            imageDetails = imageDetailsCache.get(url);
        } else {
            try {
                if (!isBlobURLCheckResultReady()) {
                    await requestBlobURLCheck();
                }
                if (awaitingForImageLoading.has(url)) {
                    const awaiters =
                        awaitingForImageLoading.get(url);
                    imageDetails = await new Promise((resolve) =>
                        awaiters.push(resolve)
                    );
                    if (!imageDetails) {
                        return null;
                    }
                } else {
                    awaitingForImageLoading.set(url, []);
                    imageDetails = await getImageDetails(url);
                    imageDetailsCache.set(url, imageDetails);
                    awaitingForImageLoading
                        .get(url)
                        .forEach((resolve) =>
                            resolve(imageDetails)
                        );
                    awaitingForImageLoading.delete(url);
                }
                if (isCancelled()) {
                    return null;
                }
            } catch (err) {
                logWarn(err);
                if (awaitingForImageLoading.has(url)) {
                    awaitingForImageLoading
                        .get(url)
                        .forEach((resolve) => resolve(null));
                    awaitingForImageLoading.delete(url);
                }
            }
        }
        if (imageDetails) {
            const bgImageValue = getBgImageValue(
                imageDetails,
                filter
            );
            if (bgImageValue) {
                return bgImageValue;
            }
        }
        if (url.startsWith("data:")) {
            const blobURL = await tryConvertDataURLToBlobURL(url);
            if (blobURL) {
                return `url("${blobURL}")`;
            }
        }
        return `url("${url}")`;
    };
};
const getBgImageValue = (imageDetails, filter) => {
    const {isDark, isLight, isTransparent, isLarge, width} =
        imageDetails;
    let result;
    const logSrc = imageDetails.src.startsWith("data:")
```

```
        ? "data:"
        : imageDetails.src;
    if (isLarge) {
        logInfo(`Not modifying too large image ${logSrc}`);
        result = null;
    } else if (
        isDark &&
        isTransparent &&
        filter.mode === 1 &&
        width > 2
    ) {
        logInfo(`Inverting dark image ${logSrc}`);
        const inverted = getFilteredImageURL(imageDetails, {
            ...filter,
            sepia: clamp(filter.sepia + 10, 0, 100)
        });
        result = `url("${inverted}")`;
    } else if (isLight && !isTransparent && filter.mode === 1) {
        logInfo(`Dimming light image ${logSrc}`);
        const dimmed = getFilteredImageURL(imageDetails, filter);
        result = `url("${dimmed}")`;
    } else if (filter.mode === 0 && isLight) {
        logInfo(`Applying filter to image ${logSrc}`);
        const filtered = getFilteredImageURL(imageDetails, {
            ...filter,
            brightness: clamp(filter.brightness - 10, 5, 200),
            sepia: clamp(filter.sepia + 10, 0, 100)
        });
        result = `url("${filtered}")`;
    } else {
        logInfo(`Not modifying the image ${logSrc}`);
        result = null;
    }
    return result;
};
const modifiers = [];
let matchIndex = 0;
let prevHasComma = false;
matches.forEach(
    ({type, match, index, typeGradient, hasComma, offset}, i) => {
        const matchStart = index;
        const prefixStart = matchIndex;
        const matchEnd = matchStart + match.length + offset;
        matchIndex = matchEnd;
        if (prefixStart !== matchStart) {
            if (prevHasComma) {
                modifiers.push(() => {
                    let betweenValue = value.substring(
                        prefixStart,
                        matchStart
                    );
                    if (betweenValue[0] === ",") {
                        betweenValue = betweenValue.substring(1);
                    }
                    return betweenValue;
                });
            } else {
                modifiers.push(() =>
                    value.substring(prefixStart, matchStart)
                );
            }
        }
        prevHasComma = hasComma || false;
        if (type === "url") {
            modifiers.push(getURLModifier(match));
        } else if (type === "gradient") {
            modifiers.push(
                getGradientModifier({
                    match,
```

```
                            index,
                            typeGradient: typeGradient,
                            hasComma: hasComma || false,
                            offset
                        })
                    );
                }
                if (i === matches.length - 1) {
                    modifiers.push(() => value.substring(matchEnd));
                }
            }
        );
        return (filter) => {
            const results = modifiers
                .filter(Boolean)
                .map((modify) => modify(filter));
            if (results.some((r) => r instanceof Promise)) {
                return Promise.all(results).then((asyncResults) => {
                    return asyncResults.filter(Boolean).join("");
                });
            }
            const combinedResult = results.join("");
            if (combinedResult.endsWith(", initial")) {
                return combinedResult.slice(0, -9);
            }
            return combinedResult;
        };
    } catch (err) {
        return null;
    }
}
function getShadowModifierWithInfo(value) {
    try {
        let index = 0;
        const colorMatches = getMatches(
            /(^|\s)(?!calc)([a-z]+\(.+?\)|#[0-9a-f]+|[a-z]+)(.*?(inset|outset)?($|,))/gi,
            value,
            2
        );
        let notParsed = 0;
        const modifiers = colorMatches.map((match, i) => {
            const prefixIndex = index;
            const matchIndex = value.indexOf(match, index);
            const matchEnd = matchIndex + match.length;
            index = matchEnd;
            const rgb = parseColorWithCache(match);
            if (!rgb) {
                notParsed++;
                return () => value.substring(prefixIndex, matchEnd);
            }
            return (filter) =>
                `${value.substring(
                    prefixIndex,
                    matchIndex
                )}${modifyShadowColor(rgb, filter)}${
                    i === colorMatches.length - 1
                        ? value.substring(matchEnd)
                        : ""
                }`;
        });
        return (filter) => {
            const modified = modifiers
                .map((modify) => modify(filter))
                .join("");
            return {
                matchesLength: colorMatches.length,
                unparseableMatchesLength: notParsed,
                result: modified
            };
```

```
        };
    } catch (err) {
        return null;
    }
}
function getShadowModifier(value) {
    const shadowModifier = getShadowModifierWithInfo(value);
    if (!shadowModifier) {
        return null;
    }
    return (theme) => shadowModifier(theme).result;
}
function getVariableModifier(
    variablesStore,
    prop,
    value,
    rule,
    ignoredImgSelectors,
    isCancelled
) {
    return variablesStore.getModifierForVariable({
        varName: prop,
        sourceValue: value,
        rule,
        ignoredImgSelectors,
        isCancelled
    });
}
function getVariableDependantModifier(variablesStore, prop, value) {
    return variablesStore.getModifierForVarDependant(prop, value);
}
function cleanModificationCache() {
    clearColorModificationCache();
    imageDetailsCache.clear();
    cleanImageProcessingCache();
    awaitingForImageLoading.clear();
}

const VAR_TYPE_BGCOLOR = 1 << 0;
const VAR_TYPE_TEXTCOLOR = 1 << 1;
const VAR_TYPE_BORDERCOLOR = 1 << 2;
const VAR_TYPE_BGIMG = 1 << 3;
class VariablesStore {
    constructor() {
        this.varTypes = new Map();
        this.rulesQueue = [];
        this.inlineStyleQueue = [];
        this.definedVars = new Set();
        this.varRefs = new Map();
        this.unknownColorVars = new Set();
        this.unknownBgVars = new Set();
        this.undefinedVars = new Set();
        this.initialVarTypes = new Map();
        this.changedTypeVars = new Set();
        this.typeChangeSubscriptions = new Map();
        this.unstableVarValues = new Map();
    }
    clear() {
        this.varTypes.clear();
        this.rulesQueue.splice(0);
        this.inlineStyleQueue.splice(0);
        this.definedVars.clear();
        this.varRefs.clear();
        this.unknownColorVars.clear();
        this.unknownBgVars.clear();
        this.undefinedVars.clear();
        this.initialVarTypes.clear();
        this.changedTypeVars.clear();
        this.typeChangeSubscriptions.clear();
```

```
    this.unstableVarValues.clear();
}
isVarType(varName, typeNum) {
    return (
        this.varTypes.has(varName) &&
        (this.varTypes.get(varName) & typeNum) > 0
    );
}
addRulesForMatching(rules) {
    this.rulesQueue.push(rules);
}
addInlineStyleForMatching(style) {
    this.inlineStyleQueue.push(style);
}
matchVariablesAndDependents() {
    if (
        this.rulesQueue.length === 0 &&
        this.inlineStyleQueue.length === 0
    ) {
        return;
    }
    this.changedTypeVars.clear();
    this.initialVarTypes = new Map(this.varTypes);
    this.collectRootVariables();
    this.collectVariablesAndVarDep();
    this.collectRootVarDependents();
    this.varRefs.forEach((refs, v) => {
        refs.forEach((r) => {
            if (this.varTypes.has(v)) {
                this.resolveVariableType(r, this.varTypes.get(v));
            }
        });
    });
    this.unknownColorVars.forEach((v) => {
        if (this.unknownBgVars.has(v)) {
            this.unknownColorVars.delete(v);
            this.unknownBgVars.delete(v);
            this.resolveVariableType(v, VAR_TYPE_BGCOLOR);
        } else if (
            this.isVarType(
                v,
                VAR_TYPE_BGCOLOR |
                    VAR_TYPE_TEXTCOLOR |
                    VAR_TYPE_BORDERCOLOR
            )
        ) {
            this.unknownColorVars.delete(v);
        } else {
            this.undefinedVars.add(v);
        }
    });
    this.unknownBgVars.forEach((v) => {
        const hasColor =
            this.findVarRef(v, (ref) => {
                return (
                    this.unknownColorVars.has(ref) ||
                    this.isVarType(
                        ref,
                        VAR_TYPE_TEXTCOLOR | VAR_TYPE_BORDERCOLOR
                    )
                );
            }) != null;
        if (hasColor) {
            this.iterateVarRefs(v, (ref) => {
                this.resolveVariableType(ref, VAR_TYPE_BGCOLOR);
            });
        } else if (
            this.isVarType(v, VAR_TYPE_BGCOLOR | VAR_TYPE_BGIMG)
        ) {
```

```
            this.unknownBgVars.delete(v);
        } else {
            this.undefinedVars.add(v);
        }
    });
    this.changedTypeVars.forEach((varName) => {
        if (this.typeChangeSubscriptions.has(varName)) {
            this.typeChangeSubscriptions
                .get(varName)
                .forEach((callback) => {
                    callback();
                });
        }
    });
    this.changedTypeVars.clear();
}
getModifierForVariable(options) {
    return (theme) => {
        const {
            varName,
            sourceValue,
            rule,
            ignoredImgSelectors,
            isCancelled
        } = options;
        const getDeclarations = () => {
            const declarations = [];
            const addModifiedValue = (
                typeNum,
                varNameWrapper,
                colorModifier
            ) => {
                if (!this.isVarType(varName, typeNum)) {
                    return;
                }
                const property = varNameWrapper(varName);
                let modifiedValue;
                if (isVarDependant(sourceValue)) {
                    if (isConstructedColorVar(sourceValue)) {
                        let value = insertVarValues(
                            sourceValue,
                            this.unstableVarValues
                        );
                        if (!value) {
                            value =
                                typeNum === VAR_TYPE_BGCOLOR
                                    ? "#ffffff"
                                    : "#000000";
                        }
                        modifiedValue = colorModifier(value, theme);
                    } else {
                        modifiedValue = replaceCSSVariablesNames(
                            sourceValue,
                            (v) => varNameWrapper(v),
                            (fallback) => colorModifier(fallback, theme)
                        );
                    }
                } else {
                    modifiedValue = colorModifier(sourceValue, theme);
                }
                declarations.push({
                    property,
                    value: modifiedValue
                });
            };
            addModifiedValue(
                VAR_TYPE_BGCOLOR,
                wrapBgColorVariableName,
                tryModifyBgColor
```

```
            );
            addModifiedValue(
                VAR_TYPE_TEXTCOLOR,
                wrapTextColorVariableName,
                tryModifyTextColor
            );
            addModifiedValue(
                VAR_TYPE_BORDERCOLOR,
                wrapBorderColorVariableName,
                tryModifyBorderColor
            );
            if (this.isVarType(varName, VAR_TYPE_BGIMG)) {
                const property = wrapBgImgVariableName(varName);
                let modifiedValue = sourceValue;
                if (isVarDependant(sourceValue)) {
                    modifiedValue = replaceCSSVariablesNames(
                        sourceValue,
                        (v) => wrapBgColorVariableName(v),
                        (fallback) => tryModifyBgColor(fallback, theme)
                    );
                }
                const bgModifier = getBgImageModifier(
                    modifiedValue,
                    rule,
                    ignoredImgSelectors,
                    isCancelled
                );
                modifiedValue =
                    typeof bgModifier === "function"
                        ? bgModifier(theme)
                        : bgModifier;
                declarations.push({
                    property,
                    value: modifiedValue
                });
            }
            return declarations;
        };
        const callbacks = new Set();
        const addListener = (onTypeChange) => {
            const callback = () => {
                const decs = getDeclarations();
                onTypeChange(decs);
            };
            callbacks.add(callback);
            this.subscribeForVarTypeChange(varName, callback);
        };
        const removeListeners = () => {
            callbacks.forEach((callback) => {
                this.unsubscribeFromVariableTypeChanges(
                    varName,
                    callback
                );
            });
        };
        return {
            declarations: getDeclarations(),
            onTypeChange: {addListener, removeListeners}
        };
    };
}
getModifierForVarDependant(property, sourceValue) {
    if (sourceValue.match(/^\s*(rgb|hsl)a?\(/)) {
        const isBg = property.startsWith("background");
        const isText = isTextColorProperty(property);
        return (theme) => {
            let value = insertVarValues(
                sourceValue,
                this.unstableVarValues
```

```
        );
        if (!value) {
            value = isBg ? "#ffffff" : "#000000";
        }
        const modifier = isBg
            ? tryModifyBgColor
            : isText
              ? tryModifyTextColor
              : tryModifyBorderColor;
        return modifier(value, theme);
    };
}
if (property === "background-color") {
    return (theme) => {
        return replaceCSSVariablesNames(
            sourceValue,
            (v) => wrapBgColorVariableName(v),
            (fallback) => tryModifyBgColor(fallback, theme)
        );
    };
}
if (isTextColorProperty(property)) {
    return (theme) => {
        return replaceCSSVariablesNames(
            sourceValue,
            (v) => wrapTextColorVariableName(v),
            (fallback) => tryModifyTextColor(fallback, theme)
        );
    };
}
if (
    property === "background" ||
    property === "background-image" ||
    property === "box-shadow"
) {
    return (theme) => {
        const unknownVars = new Set();
        const modify = () => {
            const variableReplaced = replaceCSSVariablesNames(
                sourceValue,
                (v) => {
                    if (this.isVarType(v, VAR_TYPE_BGCOLOR)) {
                        return wrapBgColorVariableName(v);
                    }
                    if (this.isVarType(v, VAR_TYPE_BGIMG)) {
                        return wrapBgImgVariableName(v);
                    }
                    unknownVars.add(v);
                    return v;
                },
                (fallback) => tryModifyBgColor(fallback, theme)
            );
            if (property === "box-shadow") {
                const shadowModifier =
                    getShadowModifierWithInfo(variableReplaced);
                const modifiedShadow = shadowModifier(theme);
                if (
                    modifiedShadow.unparseableMatchesLength !==
                    modifiedShadow.matchesLength
                ) {
                    return modifiedShadow.result;
                }
            }
            return variableReplaced;
        };
        const modified = modify();
        if (unknownVars.size > 0) {
            const isFallbackResolved = modified.match(
                /^var\(.*?, var\(--darkreader-bg--.*\)\)$/
```

```
                );
                if (isFallbackResolved) {
                    return modified;
                }
                return new Promise((resolve) => {
                    const firstUnknownVar = unknownVars
                        .values()
                        .next().value;
                    const callback = () => {
                        this.unsubscribeFromVariableTypeChanges(
                            firstUnknownVar,
                            callback
                        );
                        const newValue = modify();
                        resolve(newValue);
                    };
                    this.subscribeForVarTypeChange(
                        firstUnknownVar,
                        callback
                    );
                });
            }
            return modified;
        };
    }
    if (
        property.startsWith("border") ||
        property.startsWith("outline")
    ) {
        return (theme) => {
            return replaceCSSVariablesNames(
                sourceValue,
                (v) => wrapBorderColorVariableName(v),
                (fallback) => tryModifyBorderColor(fallback, theme)
            );
        };
    }
    return null;
}
subscribeForVarTypeChange(varName, callback) {
    if (!this.typeChangeSubscriptions.has(varName)) {
        this.typeChangeSubscriptions.set(varName, new Set());
    }
    const rootStore = this.typeChangeSubscriptions.get(varName);
    if (!rootStore.has(callback)) {
        rootStore.add(callback);
    }
}
unsubscribeFromVariableTypeChanges(varName, callback) {
    if (this.typeChangeSubscriptions.has(varName)) {
        this.typeChangeSubscriptions.get(varName).delete(callback);
    }
}
collectVariablesAndVarDep() {
    this.rulesQueue.forEach((rules) => {
        iterateCSSRules(rules, (rule) => {
            if (rule.style) {
                this.collectVarsFromCSSDeclarations(rule.style);
            }
        });
    });
    this.inlineStyleQueue.forEach((style) => {
        this.collectVarsFromCSSDeclarations(style);
    });
    this.rulesQueue.splice(0);
    this.inlineStyleQueue.splice(0);
}
collectVarsFromCSSDeclarations(style) {
    iterateCSSDeclarations(style, (property, value) => {
```

```
        if (isVariable(property)) {
            this.inspectVariable(property, value);
        }
        if (isVarDependant(value)) {
            this.inspectVarDependant(property, value);
        }
    });
}
shouldProcessRootVariables() {
    var _a;
    return (
        this.rulesQueue.length > 0 &&
        ((_a = document.documentElement.getAttribute("style")) ===
            null || _a === void 0
            ? void 0
            : _a.includes("--"))
    );
}
collectRootVariables() {
    if (!this.shouldProcessRootVariables()) {
        return;
    }
    iterateCSSDeclarations(
        document.documentElement.style,
        (property, value) => {
            if (isVariable(property)) {
                this.inspectVariable(property, value);
            }
        }
    );
}
inspectVariable(varName, value) {
    this.unstableVarValues.set(varName, value);
    if (isVarDependant(value) && isConstructedColorVar(value)) {
        this.unknownColorVars.add(varName);
        this.definedVars.add(varName);
    }
    if (this.definedVars.has(varName)) {
        return;
    }
    this.definedVars.add(varName);
    const isColor =
        rawValueRegex.test(value) || parseColorWithCache(value);
    if (isColor) {
        this.unknownColorVars.add(varName);
    } else if (
        value.includes("url(") ||
        value.includes("linear-gradient(") ||
        value.includes("radial-gradient(")
    ) {
        this.resolveVariableType(varName, VAR_TYPE_BGIMG);
    }
}
resolveVariableType(varName, typeNum) {
    const initialType = this.initialVarTypes.get(varName) || 0;
    const currentType = this.varTypes.get(varName) || 0;
    const newType = currentType | typeNum;
    this.varTypes.set(varName, newType);
    if (newType !== initialType || this.undefinedVars.has(varName)) {
        this.changedTypeVars.add(varName);
        this.undefinedVars.delete(varName);
    }
    this.unknownColorVars.delete(varName);
    this.unknownBgVars.delete(varName);
}
collectRootVarDependents() {
    if (!this.shouldProcessRootVariables()) {
        return;
    }
```

```
    iterateCSSDeclarations(
        document.documentElement.style,
        (property, value) => {
            if (isVarDependant(value)) {
                this.inspectVarDependant(property, value);
            }
        }
    );
}
inspectVarDependant(property, value) {
    if (isVariable(property)) {
        this.iterateVarDeps(value, (ref) => {
            if (!this.varRefs.has(property)) {
                this.varRefs.set(property, new Set());
            }
            this.varRefs.get(property).add(ref);
        });
    } else if (
        property === "background-color" ||
        property === "box-shadow"
    ) {
        this.iterateVarDeps(value, (v) =>
            this.resolveVariableType(v, VAR_TYPE_BGCOLOR)
        );
    } else if (isTextColorProperty(property)) {
        this.iterateVarDeps(value, (v) =>
            this.resolveVariableType(v, VAR_TYPE_TEXTCOLOR)
        );
    } else if (
        property.startsWith("border") ||
        property.startsWith("outline")
    ) {
        this.iterateVarDeps(value, (v) =>
            this.resolveVariableType(v, VAR_TYPE_BORDERCOLOR)
        );
    } else if (
        property === "background" ||
        property === "background-image"
    ) {
        this.iterateVarDeps(value, (v) => {
            if (this.isVarType(v, VAR_TYPE_BGCOLOR | VAR_TYPE_BGIMG)) {
                return;
            }
            const isBgColor =
                this.findVarRef(v, (ref) => {
                    return (
                        this.unknownColorVars.has(ref) ||
                        this.isVarType(
                            ref,
                            VAR_TYPE_TEXTCOLOR | VAR_TYPE_BORDERCOLOR
                        )
                    );
                }) != null;
            this.iterateVarRefs(v, (ref) => {
                if (isBgColor) {
                    this.resolveVariableType(ref, VAR_TYPE_BGCOLOR);
                } else {
                    this.unknownBgVars.add(ref);
                }
            });
        });
    }
}
iterateVarDeps(value, iterator) {
    const varDeps = new Set();
    iterateVarDependencies(value, (v) => varDeps.add(v));
    varDeps.forEach((v) => iterator(v));
}
findVarRef(varName, iterator, stack = new Set()) {
```

```
    if (stack.has(varName)) {
        return null;
    }
    stack.add(varName);
    const result = iterator(varName);
    if (result) {
        return varName;
    }
    const refs = this.varRefs.get(varName);
    if (!refs || refs.size === 0) {
        return null;
    }
    for (const ref of refs) {
        const found = this.findVarRef(ref, iterator, stack);
        if (found) {
            return found;
        }
    }
    return null;
}
iterateVarRefs(varName, iterator) {
    this.findVarRef(varName, (ref) => {
        iterator(ref);
        return false;
    });
}
setOnRootVariableChange(callback) {
    this.onRootVariableDefined = callback;
}
putRootVars(styleElement, theme) {
    const sheet = styleElement.sheet;
    if (sheet.cssRules.length > 0) {
        sheet.deleteRule(0);
    }
    const declarations = new Map();
    iterateCSSDeclarations(
        document.documentElement.style,
        (property, value) => {
            if (isVariable(property)) {
                if (this.isVarType(property, VAR_TYPE_BGCOLOR)) {
                    declarations.set(
                        wrapBgColorVariableName(property),
                        tryModifyBgColor(value, theme)
                    );
                }
                if (this.isVarType(property, VAR_TYPE_TEXTCOLOR)) {
                    declarations.set(
                        wrapTextColorVariableName(property),
                        tryModifyTextColor(value, theme)
                    );
                }
                if (this.isVarType(property, VAR_TYPE_BORDERCOLOR)) {
                    declarations.set(
                        wrapBorderColorVariableName(property),
                        tryModifyBorderColor(value, theme)
                    );
                }
                this.subscribeForVarTypeChange(
                    property,
                    this.onRootVariableDefined
                );
            }
        }
    );
    const cssLines = [];
    cssLines.push(":root {");
    for (const [property, value] of declarations) {
        cssLines.push(`    ${property}: ${value};`);
    }
```

```
        cssLines.push("}");
        const cssText = cssLines.join("\n");
        sheet.insertRule(cssText);
    }
}
const variablesStore = new VariablesStore();
function getVariableRange(input, searchStart = 0) {
    const start = input.indexOf("var(", searchStart);
    if (start >= 0) {
        const range = getParenthesesRange(input, start + 3);
        if (range) {
            return {start, end: range.end};
        }
    }
    return null;
}
function getVariablesMatches(input) {
    const ranges = [];
    let i = 0;
    let range;
    while ((range = getVariableRange(input, i))) {
        const {start, end} = range;
        ranges.push({start, end, value: input.substring(start, end)});
        i = range.end + 1;
    }
    return ranges;
}
function replaceVariablesMatches(input, replacer) {
    const matches = getVariablesMatches(input);
    const matchesCount = matches.length;
    if (matchesCount === 0) {
        return input;
    }
    const inputLength = input.length;
    const replacements = matches.map((m) => replacer(m.value));
    const parts = [];
    parts.push(input.substring(0, matches[0].start));
    for (let i = 0; i < matchesCount; i++) {
        parts.push(replacements[i]);
        const start = matches[i].end;
        const end =
            i < matchesCount - 1 ? matches[i + 1].start : inputLength;
        parts.push(input.substring(start, end));
    }
    return parts.join("");
}
function getVariableNameAndFallback(match) {
    const commaIndex = match.indexOf(",");
    let name;
    let fallback;
    if (commaIndex >= 0) {
        name = match.substring(4, commaIndex).trim();
        fallback = match.substring(commaIndex + 1, match.length - 1).trim();
    } else {
        name = match.substring(4, match.length - 1).trim();
        fallback = "";
    }
    return {name, fallback};
}
function replaceCSSVariablesNames(value, nameReplacer, fallbackReplacer) {
    const matchReplacer = (match) => {
        const {name, fallback} = getVariableNameAndFallback(match);
        const newName = nameReplacer(name);
        if (!fallback) {
            return `var(${newName})`;
        }
        let newFallback;
        if (isVarDependant(fallback)) {
            newFallback = replaceCSSVariablesNames(
```

```
                fallback,
                nameReplacer,
                fallbackReplacer
            );
        } else if (fallbackReplacer) {
            newFallback = fallbackReplacer(fallback);
        } else {
            newFallback = fallback;
        }
        return `var(${newName}, ${newFallback})`;
    };
    return replaceVariablesMatches(value, matchReplacer);
}
function iterateVarDependencies(value, iterator) {
    replaceCSSVariablesNames(value, (varName) => {
        iterator(varName);
        return varName;
    });
}
function wrapBgColorVariableName(name) {
    return `--darkreader-bg${name}`;
}
function wrapTextColorVariableName(name) {
    return `--darkreader-text${name}`;
}
function wrapBorderColorVariableName(name) {
    return `--darkreader-border${name}`;
}
function wrapBgImgVariableName(name) {
    return `--darkreader-bgimg${name}`;
}
function isVariable(property) {
    return property.startsWith("--");
}
function isVarDependant(value) {
    return value.includes("var(");
}
function isConstructedColorVar(value) {
    return value.match(/^\s*(rgb|hsl)a?\(/);
}
function isTextColorProperty(property) {
    return (
        property === "color" ||
        property === "caret-color" ||
        property === "-webkit-text-fill-color"
    );
}
const rawValueRegex = /^\d{1,3}, ?\d{1,3}, ?\d{1,3}$/;
function parseRawValue(color) {
    if (rawValueRegex.test(color)) {
        const splitted = color.split(",");
        let resultInRGB = "rgb(";
        splitted.forEach((number) => {
            resultInRGB += `${number.trim()}, `;
        });
        resultInRGB = resultInRGB.substring(0, resultInRGB.length - 2);
        resultInRGB += ")";
        return {isRaw: true, color: resultInRGB};
    }
    return {isRaw: false, color: color};
}
function handleRawValue(color, theme, modifyFunction) {
    const {isRaw, color: newColor} = parseRawValue(color);
    const rgb = parseColorWithCache(newColor);
    if (rgb) {
        const outputColor = modifyFunction(rgb, theme);
        if (isRaw) {
            const outputInRGB = parseColorWithCache(outputColor);
            return outputInRGB
```

```
                ? `${outputInRGB.r}, ${outputInRGB.g}, ${outputInRGB.b}`
                : outputColor;
        }
        return outputColor;
    }
    return newColor;
}
function tryModifyBgColor(color, theme) {
    return handleRawValue(color, theme, modifyBackgroundColor);
}
function tryModifyTextColor(color, theme) {
    return handleRawValue(color, theme, modifyForegroundColor);
}
function tryModifyBorderColor(color, theme) {
    return handleRawValue(color, theme, modifyBorderColor);
}
function insertVarValues(source, varValues, stack = new Set()) {
    let containsUnresolvedVar = false;
    const matchReplacer = (match) => {
        const {name, fallback} = getVariableNameAndFallback(match);
        if (stack.has(name)) {
            containsUnresolvedVar = true;
            return null;
        }
        stack.add(name);
        const varValue = varValues.get(name) || fallback;
        let inserted = null;
        if (varValue) {
            if (isVarDependant(varValue)) {
                inserted = insertVarValues(varValue, varValues, stack);
            } else {
                inserted = varValue;
            }
        }
        if (!inserted) {
            containsUnresolvedVar = true;
            return null;
        }
        return inserted;
    };
    const replaced = replaceVariablesMatches(source, matchReplacer);
    if (containsUnresolvedVar) {
        return null;
    }
    return replaced;
}

const overrides$1 = {
    "background-color": {
        customProp: "--darkreader-inline-bgcolor",
        cssProp: "background-color",
        dataAttr: "data-darkreader-inline-bgcolor"
    },
    "background-image": {
        customProp: "--darkreader-inline-bgimage",
        cssProp: "background-image",
        dataAttr: "data-darkreader-inline-bgimage"
    },
    "border-color": {
        customProp: "--darkreader-inline-border",
        cssProp: "border-color",
        dataAttr: "data-darkreader-inline-border"
    },
    "border-bottom-color": {
        customProp: "--darkreader-inline-border-bottom",
        cssProp: "border-bottom-color",
        dataAttr: "data-darkreader-inline-border-bottom"
    },
    "border-left-color": {
```

```
        customProp: "--darkreader-inline-border-left",
        cssProp: "border-left-color",
        dataAttr: "data-darkreader-inline-border-left"
    },
    "border-right-color": {
        customProp: "--darkreader-inline-border-right",
        cssProp: "border-right-color",
        dataAttr: "data-darkreader-inline-border-right"
    },
    "border-top-color": {
        customProp: "--darkreader-inline-border-top",
        cssProp: "border-top-color",
        dataAttr: "data-darkreader-inline-border-top"
    },
    "box-shadow": {
        customProp: "--darkreader-inline-boxshadow",
        cssProp: "box-shadow",
        dataAttr: "data-darkreader-inline-boxshadow"
    },
    "color": {
        customProp: "--darkreader-inline-color",
        cssProp: "color",
        dataAttr: "data-darkreader-inline-color"
    },
    "fill": {
        customProp: "--darkreader-inline-fill",
        cssProp: "fill",
        dataAttr: "data-darkreader-inline-fill"
    },
    "stroke": {
        customProp: "--darkreader-inline-stroke",
        cssProp: "stroke",
        dataAttr: "data-darkreader-inline-stroke"
    },
    "outline-color": {
        customProp: "--darkreader-inline-outline",
        cssProp: "outline-color",
        dataAttr: "data-darkreader-inline-outline"
    },
    "stop-color": {
        customProp: "--darkreader-inline-stopcolor",
        cssProp: "stop-color",
        dataAttr: "data-darkreader-inline-stopcolor"
    }
};
const shorthandOverrides = {
    background: {
        customProp: "--darkreader-inline-bg",
        cssProp: "background",
        dataAttr: "data-darkreader-inline-bg"
    }
};
const overridesList = Object.values(overrides$1);
const normalizedPropList = {};
overridesList.forEach(
    ({cssProp, customProp}) => (normalizedPropList[customProp] = cssProp)
);
const INLINE_STYLE_ATTRS = [
    "style",
    "fill",
    "stop-color",
    "stroke",
    "bgcolor",
    "color"
];
const INLINE_STYLE_SELECTOR = INLINE_STYLE_ATTRS.map(
    (attr) => `[${attr}]`
).join(", ");
function getInlineOverrideStyle() {
```

```
    const allOverrides = overridesList.concat(
        Object.values(shorthandOverrides)
    );
    return allOverrides
        .map(({dataAttr, customProp, cssProp}) => {
            return [
                `[${dataAttr}] {`,
                `  ${cssProp}: var(${customProp}) !important;`,
                "}"
            ].join("\n");
        })
        .join("\n");
}
function getInlineStyleElements(root) {
    const results = [];
    if (root instanceof Element && root.matches(INLINE_STYLE_SELECTOR)) {
        results.push(root);
    }
    if (
        root instanceof Element ||
        (isShadowDomSupported && root instanceof ShadowRoot) ||
        root instanceof Document
    ) {
        push(results, root.querySelectorAll(INLINE_STYLE_SELECTOR));
    }
    return results;
}
const treeObservers = new Map();
const attrObservers = new Map();
function watchForInlineStyles(elementStyleDidChange, shadowRootDiscovered) {
    deepWatchForInlineStyles(
        document,
        elementStyleDidChange,
        shadowRootDiscovered
    );
    iterateShadowHosts(document.documentElement, (host) => {
        deepWatchForInlineStyles(
            host.shadowRoot,
            elementStyleDidChange,
            shadowRootDiscovered
        );
    });
}
function deepWatchForInlineStyles(
    root,
    elementStyleDidChange,
    shadowRootDiscovered
) {
    if (treeObservers.has(root)) {
        treeObservers.get(root).disconnect();
        attrObservers.get(root).disconnect();
    }
    const discoveredNodes = new WeakSet();
    function discoverNodes(node) {
        getInlineStyleElements(node).forEach((el) => {
            if (discoveredNodes.has(el)) {
                return;
            }
            discoveredNodes.add(el);
            elementStyleDidChange(el);
        });
        iterateShadowHosts(node, (n) => {
            if (discoveredNodes.has(node)) {
                return;
            }
            discoveredNodes.add(node);
            shadowRootDiscovered(n.shadowRoot);
            deepWatchForInlineStyles(
                n.shadowRoot,
```

```
            elementStyleDidChange,
            shadowRootDiscovered
        );
    });
    variablesStore.matchVariablesAndDependents();
}
const treeObserver = createOptimizedTreeObserver(root, {
    onMinorMutations: ({additions}) => {
        additions.forEach((added) => discoverNodes(added));
    },
    onHugeMutations: () => {
        discoverNodes(root);
    }
});
treeObservers.set(root, treeObserver);
let attemptCount = 0;
let start = null;
const ATTEMPTS_INTERVAL = getDuration({seconds: 10});
const RETRY_TIMEOUT = getDuration({seconds: 2});
const MAX_ATTEMPTS_COUNT = 50;
let cache = [];
let timeoutId = null;
const handleAttributeMutations = throttle((mutations) => {
    const handledTargets = new Set();
    mutations.forEach((m) => {
        const target = m.target;
        if (handledTargets.has(target)) {
            return;
        }
        if (INLINE_STYLE_ATTRS.includes(m.attributeName)) {
            handledTargets.add(target);
            elementStyleDidChange(target);
        }
    });
    variablesStore.matchVariablesAndDependents();
});
const attrObserver = new MutationObserver((mutations) => {
    if (timeoutId) {
        cache.push(...mutations);
        return;
    }
    attemptCount++;
    const now = Date.now();
    if (start == null) {
        start = now;
    } else if (attemptCount >= MAX_ATTEMPTS_COUNT) {
        if (now - start < ATTEMPTS_INTERVAL) {
            timeoutId = setTimeout(() => {
                start = null;
                attemptCount = 0;
                timeoutId = null;
                const attributeCache = cache;
                cache = [];
                handleAttributeMutations(attributeCache);
            }, RETRY_TIMEOUT);
            cache.push(...mutations);
            return;
        }
        start = now;
        attemptCount = 1;
    }
    handleAttributeMutations(mutations);
});
attrObserver.observe(root, {
    attributes: true,
    attributeFilter: INLINE_STYLE_ATTRS.concat(
        overridesList.map(({dataAttr}) => dataAttr)
    ),
    subtree: true
```

```
    });
    attrObservers.set(root, attrObserver);
}
function stopWatchingForInlineStyles() {
    treeObservers.forEach((o) => o.disconnect());
    attrObservers.forEach((o) => o.disconnect());
    treeObservers.clear();
    attrObservers.clear();
}
const inlineStyleCache = new WeakMap();
const filterProps = [
    "brightness",
    "contrast",
    "grayscale",
    "sepia",
    "mode"
];
function getInlineStyleCacheKey(el, theme) {
    return INLINE_STYLE_ATTRS.map(
        (attr) => `${attr}="${el.getAttribute(attr)}"`
    )
        .concat(filterProps.map((prop) => `${prop}="${theme[prop]}"`))
        .join(" ");
}
function shouldIgnoreInlineStyle(element, selectors) {
    for (let i = 0, len = selectors.length; i < len; i++) {
        const ingnoredSelector = selectors[i];
        if (element.matches(ingnoredSelector)) {
            return true;
        }
    }
    return false;
}
function overrideInlineStyle(
    element,
    theme,
    ignoreInlineSelectors,
    ignoreImageSelectors
) {
    var _a;
    const cacheKey = getInlineStyleCacheKey(element, theme);
    if (cacheKey === inlineStyleCache.get(element)) {
        return;
    }
    const unsetProps = new Set(Object.keys(overrides$1));
    function setCustomProp(targetCSSProp, modifierCSSProp, cssVal) {
        const mod = getModifiableCSSDeclaration(
            modifierCSSProp,
            cssVal,
            {style: element.style},
            variablesStore,
            ignoreImageSelectors,
            null
        );
        if (!mod) {
            return;
        }
        function setStaticValue(value) {
            var _a;
            const {customProp, dataAttr} =
                (_a = overrides$1[targetCSSProp]) !== null && _a !== void 0
                    ? _a
                    : shorthandOverrides[targetCSSProp];
            element.style.setProperty(customProp, value);
            if (!element.hasAttribute(dataAttr)) {
                element.setAttribute(dataAttr, "");
            }
            unsetProps.delete(targetCSSProp);
        }
```

```
    function setVarDeclaration(mod) {
        let prevDeclarations = [];
        function setProps(declarations) {
            prevDeclarations.forEach(({property}) => {
                element.style.removeProperty(property);
            });
            declarations.forEach(({property, value}) => {
                if (!(value instanceof Promise)) {
                    element.style.setProperty(property, value);
                }
            });
            prevDeclarations = declarations;
        }
        setProps(mod.declarations);
        mod.onTypeChange.addListener(setProps);
    }
    function setAsyncValue(promise) {
        promise.then((value) => {
            if (
                value &&
                targetCSSProp === "background" &&
                value.startsWith("var(--darkreader-bg--")
            ) {
                setStaticValue(value);
            }
        });
    }
    const value =
        typeof mod.value === "function" ? mod.value(theme) : mod.value;
    if (typeof value === "string") {
        setStaticValue(value);
    } else if (value instanceof Promise) {
        setAsyncValue(value);
    } else if (typeof value === "object") {
        setVarDeclaration(value);
    }
}
if (ignoreInlineSelectors.length > 0) {
    if (shouldIgnoreInlineStyle(element, ignoreInlineSelectors)) {
        unsetProps.forEach((cssProp) => {
            element.removeAttribute(overrides$1[cssProp].dataAttr);
        });
        return;
    }
}
if (element.hasAttribute("bgcolor")) {
    let value = element.getAttribute("bgcolor");
    if (
        value.match(/^[0-9a-f]{3}$/i) ||
        value.match(/^[0-9a-f]{6}$/i)
    ) {
        value = `#${value}`;
    }
    setCustomProp("background-color", "background-color", value);
}
if (element.hasAttribute("color") && element.rel !== "mask-icon") {
    let value = element.getAttribute("color");
    if (
        value.match(/^[0-9a-f]{3}$/i) ||
        value.match(/^[0-9a-f]{6}$/i)
    ) {
        value = `#${value}`;
    }
    setCustomProp("color", "color", value);
}
if (element instanceof SVGElement) {
    if (element.hasAttribute("fill")) {
        const SMALL_SVG_LIMIT = 32;
        const value = element.getAttribute("fill");
```

```
        if (value !== "none") {
            if (!(element instanceof SVGTextElement)) {
                const handleSVGElement = () => {
                    const {width, height} =
                        element.getBoundingClientRect();
                    const isBg =
                        width > SMALL_SVG_LIMIT ||
                        height > SMALL_SVG_LIMIT;
                    setCustomProp(
                        "fill",
                        isBg ? "background-color" : "color",
                        value
                    );
                };
                if (isReadyStateComplete()) {
                    handleSVGElement();
                } else {
                    addReadyStateCompleteListener(handleSVGElement);
                }
            } else {
                setCustomProp("fill", "color", value);
            }
        }
    }
    if (element.hasAttribute("stop-color")) {
        setCustomProp(
            "stop-color",
            "background-color",
            element.getAttribute("stop-color")
        );
    }
}
if (element.hasAttribute("stroke")) {
    const value = element.getAttribute("stroke");
    setCustomProp(
        "stroke",
        element instanceof SVGLineElement ||
            element instanceof SVGTextElement
            ? "border-color"
            : "color",
        value
    );
}
element.style &&
    iterateCSSDeclarations(element.style, (property, value) => {
        if (property === "background-image" && value.includes("url")) {
            return;
        }
        if (
            overrides$1.hasOwnProperty(property) ||
            (property.startsWith("--") && !normalizedPropList[property])
        ) {
            setCustomProp(property, property, value);
        } else if (
            property === "background" &&
            value.includes("var(")
        ) {
            setCustomProp("background", "background", value);
        } else {
            const overriddenProp = normalizedPropList[property];
            if (
                overriddenProp &&
                !element.style.getPropertyValue(overriddenProp) &&
                !element.hasAttribute(overriddenProp)
            ) {
                if (
                    overriddenProp === "background-color" &&
                    element.hasAttribute("bgcolor")
                ) {
```

```
                    return;
                }
                element.style.setProperty(property, "");
            }
        }
    });
    if (
        element.style &&
        element instanceof SVGTextElement &&
        element.style.fill
    ) {
        setCustomProp(
            "fill",
            "color",
            element.style.getPropertyValue("fill")
        );
    }
    if (
        (_a = element.getAttribute("style")) === null || _a === void 0
            ? void 0
            : _a.includes("--")
    ) {
        variablesStore.addInlineStyleForMatching(element.style);
    }
    forEach(unsetProps, (cssProp) => {
        element.removeAttribute(overrides$1[cssProp].dataAttr);
    });
    inlineStyleCache.set(element, getInlineStyleCacheKey(element, theme));
}

const metaThemeColorName = "theme-color";
const metaThemeColorSelector = `meta[name="${metaThemeColorName}"]`;
let srcMetaThemeColor = null;
let observer = null;
function changeMetaThemeColor(meta, theme) {
    srcMetaThemeColor = srcMetaThemeColor || meta.content;
    const color = parseColorWithCache(srcMetaThemeColor);
    if (!color) {
        return;
    }
    meta.content = modifyBackgroundColor(color, theme);
}
function changeMetaThemeColorWhenAvailable(theme) {
    const meta = document.querySelector(metaThemeColorSelector);
    if (meta) {
        changeMetaThemeColor(meta, theme);
    } else {
        if (observer) {
            observer.disconnect();
        }
        observer = new MutationObserver((mutations) => {
            loop: for (let i = 0; i < mutations.length; i++) {
                const {addedNodes} = mutations[i];
                for (let j = 0; j < addedNodes.length; j++) {
                    const node = addedNodes[j];
                    if (
                        node instanceof HTMLMetaElement &&
                        node.name === metaThemeColorName
                    ) {
                        observer.disconnect();
                        observer = null;
                        changeMetaThemeColor(node, theme);
                        break loop;
                    }
                }
            }
        });
        observer.observe(document.head, {childList: true});
    }
```

```
}
function restoreMetaThemeColor() {
    if (observer) {
        observer.disconnect();
        observer = null;
    }
    const meta = document.querySelector(metaThemeColorSelector);
    if (meta && srcMetaThemeColor) {
        meta.content = srcMetaThemeColor;
    }
}

const themeCacheKeys = [
    "mode",
    "brightness",
    "contrast",
    "grayscale",
    "sepia",
    "darkSchemeBackgroundColor",
    "darkSchemeTextColor",
    "lightSchemeBackgroundColor",
    "lightSchemeTextColor"
];
function getThemeKey(theme) {
    let resultKey = "";
    themeCacheKeys.forEach((key) => {
        resultKey += `${key}:${theme[key]};`;
    });
    return resultKey;
}
const asyncQueue = createAsyncTasksQueue();
function createStyleSheetModifier() {
    let renderId = 0;
    const rulesTextCache = new Set();
    const rulesModCache = new Map();
    const varTypeChangeCleaners = new Set();
    let prevFilterKey = null;
    let hasNonLoadedLink = false;
    let wasRebuilt = false;
    function shouldRebuildStyle() {
        return hasNonLoadedLink && !wasRebuilt;
    }
    function modifySheet(options) {
        const rules = options.sourceCSSRules;
        const {
            theme,
            ignoreImageAnalysis,
            force,
            prepareSheet,
            isAsyncCancelled
        } = options;
        let rulesChanged = rulesModCache.size === 0;
        const notFoundCacheKeys = new Set(rulesModCache.keys());
        const themeKey = getThemeKey(theme);
        const themeChanged = themeKey !== prevFilterKey;
        if (hasNonLoadedLink) {
            wasRebuilt = true;
        }
        const modRules = [];
        iterateCSSRules(
            rules,
            (rule) => {
                let cssText = rule.cssText;
                let textDiffersFromPrev = false;
                notFoundCacheKeys.delete(cssText);
                if (rule.parentRule instanceof CSSMediaRule) {
                    cssText += `;${rule.parentRule.media.mediaText}`;
                }
                if (!rulesTextCache.has(cssText)) {
```

```
            rulesTextCache.add(cssText);
            textDiffersFromPrev = true;
        }
        if (textDiffersFromPrev) {
            rulesChanged = true;
        } else {
            modRules.push(rulesModCache.get(cssText));
            return;
        }
        if (rule.style.all === "revert") {
            return;
        }
        const modDecs = [];
        rule.style &&
            iterateCSSDeclarations(
                rule.style,
                (property, value) => {
                    const mod = getModifiableCSSDeclaration(
                        property,
                        value,
                        rule,
                        variablesStore,
                        ignoreImageAnalysis,
                        isAsyncCancelled
                    );
                    if (mod) {
                        modDecs.push(mod);
                    }
                }
            );
        let modRule = null;
        if (modDecs.length > 0) {
            const parentRule = rule.parentRule;
            modRule = {
                selector: rule.selectorText,
                declarations: modDecs,
                parentRule
            };
            modRules.push(modRule);
        }
        rulesModCache.set(cssText, modRule);
    },
    () => {
        hasNonLoadedLink = true;
    }
);
notFoundCacheKeys.forEach((key) => {
    rulesTextCache.delete(key);
    rulesModCache.delete(key);
});
prevFilterKey = themeKey;
if (!force && !rulesChanged && !themeChanged) {
    return;
}
renderId++;
function setRule(target, index, rule) {
    const {selector, declarations} = rule;
    let selectorText = selector;
    if (
        selector.startsWith(":is(") &&
        (selector.includes(":is()") ||
            selector.includes(":where()") ||
            (selector.includes(":where(") &&
                selector.includes(":-moz")))
    ) {
        selectorText = ".darkreader-unsupported-selector";
    }
    let ruleText = `${selectorText} {`;
    for (const dec of declarations) {
```

```
        const {property, value, important} = dec;
        if (value) {
            ruleText += ` ${property}: ${value}${
                important ? " !important" : ""
            };`;
        }
    }
    ruleText += " }";
    target.insertRule(ruleText, index);
}
const asyncDeclarations = new Map();
const varDeclarations = new Map();
let asyncDeclarationCounter = 0;
let varDeclarationCounter = 0;
const rootReadyGroup = {rule: null, rules: [], isGroup: true};
const groupRefs = new WeakMap();
function getGroup(rule) {
    if (rule == null) {
        return rootReadyGroup;
    }
    if (groupRefs.has(rule)) {
        return groupRefs.get(rule);
    }
    const group = {rule, rules: [], isGroup: true};
    groupRefs.set(rule, group);
    const parentGroup = getGroup(rule.parentRule);
    parentGroup.rules.push(group);
    return group;
}
varTypeChangeCleaners.forEach((clear) => clear());
varTypeChangeCleaners.clear();
modRules
    .filter((r) => r)
    .forEach(({selector, declarations, parentRule}) => {
        const group = getGroup(parentRule);
        const readyStyleRule = {
            selector,
            declarations: [],
            isGroup: false
        };
        const readyDeclarations = readyStyleRule.declarations;
        group.rules.push(readyStyleRule);
        function handleAsyncDeclaration(
            property,
            modified,
            important,
            sourceValue
        ) {
            const asyncKey = ++asyncDeclarationCounter;
            const asyncDeclaration = {
                property,
                value: null,
                important,
                asyncKey,
                sourceValue
            };
            readyDeclarations.push(asyncDeclaration);
            const currentRenderId = renderId;
            modified.then((asyncValue) => {
                if (
                    !asyncValue ||
                    isAsyncCancelled() ||
                    currentRenderId !== renderId
                ) {
                    return;
                }
                asyncDeclaration.value = asyncValue;
                asyncQueue.add(() => {
                    if (
```

```
                isAsyncCancelled() ||
                currentRenderId !== renderId
            ) {
                return;
            }
            rebuildAsyncRule(asyncKey);
        });
    });
}
function handleVarDeclarations(
    property,
    modified,
    important,
    sourceValue
) {
    const {declarations: varDecs, onTypeChange} = modified;
    const varKey = ++varDeclarationCounter;
    const currentRenderId = renderId;
    const initialIndex = readyDeclarations.length;
    let oldDecs = [];
    if (varDecs.length === 0) {
        const tempDec = {
            property,
            value: sourceValue,
            important,
            sourceValue,
            varKey
        };
        readyDeclarations.push(tempDec);
        oldDecs = [tempDec];
    }
    varDecs.forEach((mod) => {
        if (mod.value instanceof Promise) {
            handleAsyncDeclaration(
                mod.property,
                mod.value,
                important,
                sourceValue
            );
        } else {
            const readyDec = {
                property: mod.property,
                value: mod.value,
                important,
                sourceValue,
                varKey
            };
            readyDeclarations.push(readyDec);
            oldDecs.push(readyDec);
        }
    });
    onTypeChange.addListener((newDecs) => {
        if (
            isAsyncCancelled() ||
            currentRenderId !== renderId
        ) {
            return;
        }
        const readyVarDecs = newDecs.map((mod) => {
            return {
                property: mod.property,
                value: mod.value,
                important,
                sourceValue,
                varKey
            };
        });
        const index = readyDeclarations.indexOf(
            oldDecs[0],
```

```
                    initialIndex
                );
                readyDeclarations.splice(
                    index,
                    oldDecs.length,
                    ...readyVarDecs
                );
                oldDecs = readyVarDecs;
                rebuildVarRule(varKey);
            });
            varTypeChangeCleaners.add(() =>
                onTypeChange.removeListeners()
            );
        }
        declarations.forEach(
            ({property, value, important, sourceValue}) => {
                if (typeof value === "function") {
                    const modified = value(theme);
                    if (modified instanceof Promise) {
                        handleAsyncDeclaration(
                            property,
                            modified,
                            important,
                            sourceValue
                        );
                    } else if (property.startsWith("--")) {
                        handleVarDeclarations(
                            property,
                            modified,
                            important,
                            sourceValue
                        );
                    } else {
                        readyDeclarations.push({
                            property,
                            value: modified,
                            important,
                            sourceValue
                        });
                    }
                } else {
                    readyDeclarations.push({
                        property,
                        value,
                        important,
                        sourceValue
                    });
                }
            }
        );
    });
const sheet = prepareSheet();
function buildStyleSheet() {
    function createTarget(group, parent) {
        const {rule} = group;
        if (rule instanceof CSSMediaRule) {
            const {media} = rule;
            const index = parent.cssRules.length;
            parent.insertRule(
                `@media ${media.mediaText} {}`,
                index
            );
            return parent.cssRules[index];
        }
        return parent;
    }
    function iterateReadyRules(group, target, styleIterator) {
        group.rules.forEach((r) => {
            if (r.isGroup) {
```

```
                        const t = createTarget(r, target);
                        iterateReadyRules(r, t, styleIterator);
                    } else {
                        styleIterator(r, target);
                    }
                });
            }
            iterateReadyRules(rootReadyGroup, sheet, (rule, target) => {
                const index = target.cssRules.length;
                rule.declarations.forEach(({asyncKey, varKey}) => {
                    if (asyncKey != null) {
                        asyncDeclarations.set(asyncKey, {
                            rule,
                            target,
                            index
                        });
                    }
                    if (varKey != null) {
                        varDeclarations.set(varKey, {rule, target, index});
                    }
                });
                setRule(target, index, rule);
            });
        }
        function rebuildAsyncRule(key) {
            const {rule, target, index} = asyncDeclarations.get(key);
            target.deleteRule(index);
            setRule(target, index, rule);
            asyncDeclarations.delete(key);
        }
        function rebuildVarRule(key) {
            const {rule, target, index} = varDeclarations.get(key);
            target.deleteRule(index);
            setRule(target, index, rule);
        }
        buildStyleSheet();
    }
    return {modifySheet, shouldRebuildStyle};
}

const STYLE_SELECTOR = 'style, link[rel*="stylesheet" i]:not([disabled])';
function isFontsGoogleApiStyle(element) {
    if (!element.href) {
        return false;
    }
    try {
        const elementURL = new URL(element.href);
        return elementURL.hostname === "fonts.googleapis.com";
    } catch (err) {
        logInfo(`Couldn't construct ${element.href} as URL`);
        return false;
    }
}
const hostsBreakingOnSVGStyleOverride = ["www.onet.pl"];
function shouldManageStyle(element) {
    return (
        (element instanceof HTMLStyleElement ||
            (element instanceof SVGStyleElement &&
                !hostsBreakingOnSVGStyleOverride.includes(
                    location.hostname
                )) ||
            (element instanceof HTMLLinkElement &&
                Boolean(element.rel) &&
                element.rel.toLowerCase().includes("stylesheet") &&
                Boolean(element.href) &&
                !element.disabled &&
                true &&
                !isFontsGoogleApiStyle(element))) &&
        !element.classList.contains("darkreader") &&
```

```
        element.media.toLowerCase() !== "print" &&
        !element.classList.contains("stylus")
    );
}
function getManageableStyles(node, results = [], deep = true) {
    if (shouldManageStyle(node)) {
        results.push(node);
    } else if (
        node instanceof Element ||
        (isShadowDomSupported && node instanceof ShadowRoot) ||
        node === document
    ) {
        forEach(node.querySelectorAll(STYLE_SELECTOR), (style) =>
            getManageableStyles(style, results, false)
        );
        if (deep) {
            iterateShadowHosts(node, (host) =>
                getManageableStyles(host.shadowRoot, results, false)
            );
        }
    }
    return results;
}
const syncStyleSet = new WeakSet();
const corsStyleSet = new WeakSet();
let canOptimizeUsingProxy$1 = false;
document.addEventListener(
    "__darkreader__inlineScriptsAllowed",
    () => {
        canOptimizeUsingProxy$1 = true;
    },
    {once: true, passive: true}
);
let loadingLinkCounter = 0;
const rejectorsForLoadingLinks = new Map();
function cleanLoadingLinks() {
    rejectorsForLoadingLinks.clear();
}
function manageStyle(element, {update, loadingStart, loadingEnd}) {
    const prevStyles = [];
    let next = element;
    while (
        (next = next.nextElementSibling) &&
        next.matches(".darkreader")
    ) {
        prevStyles.push(next);
    }
    let corsCopy =
        prevStyles.find(
            (el) => el.matches(".darkreader--cors") && !corsStyleSet.has(el)
        ) || null;
    let syncStyle =
        prevStyles.find(
            (el) => el.matches(".darkreader--sync") && !syncStyleSet.has(el)
        ) || null;
    let corsCopyPositionWatcher = null;
    let syncStylePositionWatcher = null;
    let cancelAsyncOperations = false;
    let isOverrideEmpty = true;
    const sheetModifier = createStyleSheetModifier();
    const observer = new MutationObserver(() => {
        update();
    });
    const observerOptions = {
        attributes: true,
        childList: true,
        subtree: true,
        characterData: true
    };
```

```
function containsCSSImport() {
    var _a;
    if (!(element instanceof HTMLStyleElement)) {
        return false;
    }
    const cssText = removeCSSComments(
        (_a = element.textContent) !== null && _a !== void 0 ? _a : ""
    ).trim();
    return cssText.match(cssImportRegex);
}
function hasImports(cssRules, checkCrossOrigin) {
    let result = false;
    if (cssRules) {
        let rule;
        cssRulesLoop: for (
            let i = 0, len = cssRules.length;
            i < len;
            i++
        ) {
            rule = cssRules[i];
            if (rule.href) {
                if (checkCrossOrigin) {
                    if (
                        !rule.href.startsWith(
                            "https://fonts.googleapis.com/"
                        ) &&
                        rule.href.startsWith("http") &&
                        !rule.href.startsWith(location.origin)
                    ) {
                        result = true;
                        break cssRulesLoop;
                    }
                } else {
                    result = true;
                    break cssRulesLoop;
                }
            }
        }
    }
    return result;
}
function getRulesSync() {
    if (corsCopy) {
        return corsCopy.sheet.cssRules;
    }
    if (containsCSSImport()) {
        return null;
    }
    const cssRules = safeGetSheetRules();
    if (
        element instanceof HTMLLinkElement &&
        !isRelativeHrefOnAbsolutePath(element.href) &&
        hasImports(cssRules, false)
    ) {
        return null;
    }
    if (hasImports(cssRules, true)) {
        return null;
    }
    return cssRules;
}
function insertStyle() {
    if (corsCopy) {
        if (element.nextSibling !== corsCopy) {
            element.parentNode.insertBefore(
                corsCopy,
                element.nextSibling
            );
        }
```

```
        if (corsCopy.nextSibling !== syncStyle) {
            element.parentNode.insertBefore(
                syncStyle,
                corsCopy.nextSibling
            );
        }
    } else if (element.nextSibling !== syncStyle) {
        element.parentNode.insertBefore(syncStyle, element.nextSibling);
    }
}
function createSyncStyle() {
    syncStyle =
        element instanceof SVGStyleElement
            ? document.createElementNS(
                  "http://www.w3.org/2000/svg",
                  "style"
              )
            : document.createElement("style");
    syncStyle.classList.add("darkreader");
    syncStyle.classList.add("darkreader--sync");
    syncStyle.media = "screen";
    if (element.title) {
        syncStyle.title = element.title;
    }
    syncStyleSet.add(syncStyle);
}
let isLoadingRules = false;
let wasLoadingError = false;
const loadingLinkId = ++loadingLinkCounter;
async function getRulesAsync() {
    let cssText;
    let cssBasePath;
    if (element instanceof HTMLLinkElement) {
        let [cssRules, accessError] = getRulesOrError();
        if (
            (!cssRules && !accessError) ||
            isStillLoadingError(accessError)
        ) {
            try {
                logInfo(
                    `Linkelement ${loadingLinkId} is not loaded yet and thus will be await for`,
                    element
                );
                await linkLoading(element, loadingLinkId);
            } catch (err) {
                wasLoadingError = true;
            }
            if (cancelAsyncOperations) {
                return null;
            }
            [cssRules, accessError] = getRulesOrError();
        }
        if (cssRules) {
            if (!hasImports(cssRules, false)) {
                return cssRules;
            }
        }
        cssText = await loadText(element.href);
        cssBasePath = getCSSBaseBath(element.href);
        if (cancelAsyncOperations) {
            return null;
        }
    } else if (containsCSSImport()) {
        cssText = element.textContent.trim();
        cssBasePath = getCSSBaseBath(location.href);
    } else {
        return null;
    }
    if (cssText) {
```

```
        try {
            const fullCSSText = await replaceCSSImports(
                cssText,
                cssBasePath
            );
            corsCopy = createCORSCopy(element, fullCSSText);
        } catch (err) {}
        if (corsCopy) {
            corsCopyPositionWatcher = watchForNodePosition(
                corsCopy,
                "prev-sibling"
            );
            return corsCopy.sheet.cssRules;
        }
    }
    return null;
}
function details(options) {
    const rules = getRulesSync();
    if (!rules) {
        if (options.secondRound) {
            return null;
        }
        if (isLoadingRules || wasLoadingError) {
            return null;
        }
        isLoadingRules = true;
        loadingStart();
        getRulesAsync()
            .then((results) => {
                isLoadingRules = false;
                loadingEnd();
                if (results) {
                    update();
                }
            })
            .catch((err) => {
                isLoadingRules = false;
                loadingEnd();
            });
        return null;
    }
    return {rules};
}
let forceRenderStyle = false;
function render(theme, ignoreImageAnalysis) {
    const rules = getRulesSync();
    if (!rules) {
        return;
    }
    cancelAsyncOperations = false;
    function removeCSSRulesFromSheet(sheet) {
        if (!sheet) {
            return;
        }
        for (let i = sheet.cssRules.length - 1; i >= 0; i--) {
            sheet.deleteRule(i);
        }
    }
    function prepareOverridesSheet() {
        if (!syncStyle) {
            createSyncStyle();
        }
        syncStylePositionWatcher && syncStylePositionWatcher.stop();
        insertStyle();
        if (syncStyle.sheet == null) {
            syncStyle.textContent = "";
        }
        const sheet = syncStyle.sheet;
```

```
        removeCSSRulesFromSheet(sheet);
        if (syncStylePositionWatcher) {
            syncStylePositionWatcher.run();
        } else {
            syncStylePositionWatcher = watchForNodePosition(
                syncStyle,
                "prev-sibling",
                () => {
                    forceRenderStyle = true;
                    buildOverrides();
                }
            );
        }
        return syncStyle.sheet;
    }
    function buildOverrides() {
        const force = forceRenderStyle;
        forceRenderStyle = false;
        sheetModifier.modifySheet({
            prepareSheet: prepareOverridesSheet,
            sourceCSSRules: rules,
            theme,
            ignoreImageAnalysis,
            force,
            isAsyncCancelled: () => cancelAsyncOperations
        });
        isOverrideEmpty = syncStyle.sheet.cssRules.length === 0;
        if (sheetModifier.shouldRebuildStyle()) {
            addReadyStateCompleteListener(() => update());
        }
    }
    buildOverrides();
}
function getRulesOrError() {
    try {
        if (element.sheet == null) {
            return [null, null];
        }
        return [element.sheet.cssRules, null];
    } catch (err) {
        return [null, err];
    }
}
function isStillLoadingError(error) {
    return error && error.message && error.message.includes("loading");
}
function safeGetSheetRules() {
    const [cssRules, err] = getRulesOrError();
    if (err) {
        return null;
    }
    return cssRules;
}
function watchForSheetChanges() {
    watchForSheetChangesUsingProxy();
    if (!(canOptimizeUsingProxy$1 && element.sheet)) {
        watchForSheetChangesUsingRAF();
    }
}
let rulesChangeKey = null;
let rulesCheckFrameId = null;
function getRulesChangeKey() {
    const rules = safeGetSheetRules();
    return rules ? rules.length : null;
}
function didRulesKeyChange() {
    return getRulesChangeKey() !== rulesChangeKey;
}
function watchForSheetChangesUsingRAF() {
```

```
    rulesChangeKey = getRulesChangeKey();
    stopWatchingForSheetChangesUsingRAF();
    const checkForUpdate = () => {
        if (didRulesKeyChange()) {
            rulesChangeKey = getRulesChangeKey();
            update();
        }
        if (canOptimizeUsingProxy$1 && element.sheet) {
            stopWatchingForSheetChangesUsingRAF();
            return;
        }
        rulesCheckFrameId = requestAnimationFrame(checkForUpdate);
    };
    checkForUpdate();
}
function stopWatchingForSheetChangesUsingRAF() {
    cancelAnimationFrame(rulesCheckFrameId);
}
let areSheetChangesPending = false;
function onSheetChange() {
    canOptimizeUsingProxy$1 = true;
    stopWatchingForSheetChangesUsingRAF();
    if (areSheetChangesPending) {
        return;
    }
    function handleSheetChanges() {
        areSheetChangesPending = false;
        if (cancelAsyncOperations) {
            return;
        }
        update();
    }
    areSheetChangesPending = true;
    if (typeof queueMicrotask === "function") {
        queueMicrotask(handleSheetChanges);
    } else {
        requestAnimationFrame(handleSheetChanges);
    }
}
function watchForSheetChangesUsingProxy() {
    element.addEventListener(
        "__darkreader__updateSheet",
        onSheetChange,
        {passive: true}
    );
}
function stopWatchingForSheetChangesUsingProxy() {
    element.removeEventListener(
        "__darkreader__updateSheet",
        onSheetChange
    );
}
function stopWatchingForSheetChanges() {
    stopWatchingForSheetChangesUsingProxy();
    stopWatchingForSheetChangesUsingRAF();
}
function pause() {
    observer.disconnect();
    cancelAsyncOperations = true;
    corsCopyPositionWatcher && corsCopyPositionWatcher.stop();
    syncStylePositionWatcher && syncStylePositionWatcher.stop();
    stopWatchingForSheetChanges();
}
function destroy() {
    pause();
    removeNode(corsCopy);
    removeNode(syncStyle);
    loadingEnd();
    if (rejectorsForLoadingLinks.has(loadingLinkId)) {
```

```
            const reject = rejectorsForLoadingLinks.get(loadingLinkId);
            rejectorsForLoadingLinks.delete(loadingLinkId);
            reject && reject();
        }
    }
    function watch() {
        observer.observe(element, observerOptions);
        if (element instanceof HTMLStyleElement) {
            watchForSheetChanges();
        }
    }
    const maxMoveCount = 10;
    let moveCount = 0;
    function restore() {
        if (!syncStyle) {
            return;
        }
        moveCount++;
        if (moveCount > maxMoveCount) {
            return;
        }
        insertStyle();
        corsCopyPositionWatcher && corsCopyPositionWatcher.skip();
        syncStylePositionWatcher && syncStylePositionWatcher.skip();
        if (!isOverrideEmpty) {
            forceRenderStyle = true;
            update();
        }
    }
    return {
        details,
        render,
        pause,
        destroy,
        watch,
        restore
    };
}
async function linkLoading(link, loadingId) {
    return new Promise((resolve, reject) => {
        const cleanUp = () => {
            link.removeEventListener("load", onLoad);
            link.removeEventListener("error", onError);
            rejectorsForLoadingLinks.delete(loadingId);
        };
        const onLoad = () => {
            cleanUp();
            resolve();
        };
        const onError = () => {
            cleanUp();
            reject(
                `Linkelement ${loadingId} couldn't be loaded. ${link.href}`
            );
        };
        rejectorsForLoadingLinks.set(loadingId, () => {
            cleanUp();
            reject();
        });
        link.addEventListener("load", onLoad, {passive: true});
        link.addEventListener("error", onError, {passive: true});
        if (!link.href) {
            onError();
        }
    });
}
function getCSSImportURL(importDeclaration) {
    return getCSSURLValue(
        importDeclaration
```

```
            .substring(7)
            .trim()
            .replace(/;$/, "")
            .replace(/screen$/, "")
    );
}
async function loadText(url) {
    if (url.startsWith("data:")) {
        return await (await fetch(url)).text();
    }
    return await bgFetch({
        url,
        responseType: "text",
        mimeType: "text/css",
        origin: window.location.origin
    });
}
async function replaceCSSImports(cssText, basePath, cache = new Map()) {
    cssText = removeCSSComments(cssText);
    cssText = replaceCSSFontFace(cssText);
    cssText = replaceCSSRelativeURLsWithAbsolute(cssText, basePath);
    const importMatches = getMatches(cssImportRegex, cssText);
    for (const match of importMatches) {
        const importURL = getCSSImportURL(match);
        const absoluteURL = getAbsoluteURL(basePath, importURL);
        let importedCSS;
        if (cache.has(absoluteURL)) {
            importedCSS = cache.get(absoluteURL);
        } else {
            try {
                importedCSS = await loadText(absoluteURL);
                cache.set(absoluteURL, importedCSS);
                importedCSS = await replaceCSSImports(
                    importedCSS,
                    getCSSBaseBath(absoluteURL),
                    cache
                );
            } catch (err) {
                importedCSS = "";
            }
        }
        cssText = cssText.split(match).join(importedCSS);
    }
    cssText = cssText.trim();
    return cssText;
}
function createCORSCopy(srcElement, cssText) {
    if (!cssText) {
        return null;
    }
    const cors = document.createElement("style");
    cors.classList.add("darkreader");
    cors.classList.add("darkreader--cors");
    cors.media = "screen";
    cors.textContent = cssText;
    srcElement.parentNode.insertBefore(cors, srcElement.nextSibling);
    cors.sheet.disabled = true;
    corsStyleSet.add(cors);
    return cors;
}

const observers = [];
let observedRoots;
const definedCustomElements = new Set();
const undefinedGroups = new Map();
let elementsDefinitionCallback;
function isCustomElement(element) {
    if (element.tagName.includes("-") || element.getAttribute("is")) {
        return true;
```

```
    }
    return false;
}
function recordUndefinedElement(element) {
    let tag = element.tagName.toLowerCase();
    if (!tag.includes("-")) {
        const extendedTag = element.getAttribute("is");
        if (extendedTag) {
            tag = extendedTag;
        } else {
            return;
        }
    }
    if (!undefinedGroups.has(tag)) {
        undefinedGroups.set(tag, new Set());
        customElementsWhenDefined(tag).then(() => {
            if (elementsDefinitionCallback) {
                const elements = undefinedGroups.get(tag);
                undefinedGroups.delete(tag);
                elementsDefinitionCallback(Array.from(elements));
            }
        });
    }
    undefinedGroups.get(tag).add(element);
}
function collectUndefinedElements(root) {
    if (!isDefinedSelectorSupported) {
        return;
    }
    forEach(
        root.querySelectorAll(":not(:defined)"),
        recordUndefinedElement
    );
}
let canOptimizeUsingProxy = false;
document.addEventListener(
    "__darkreader__inlineScriptsAllowed",
    () => {
        canOptimizeUsingProxy = true;
    },
    {once: true, passive: true}
);
const resolvers = new Map();
function handleIsDefined(e) {
    canOptimizeUsingProxy = true;
    const tag = e.detail.tag;
    definedCustomElements.add(tag);
    if (resolvers.has(tag)) {
        const r = resolvers.get(tag);
        resolvers.delete(tag);
        r.forEach((r) => r());
    }
}
async function customElementsWhenDefined(tag) {
    if (definedCustomElements.has(tag)) {
        return;
    }
    return new Promise((resolve) => {
        if (
            window.customElements &&
            typeof customElements.whenDefined === "function"
        ) {
            customElements.whenDefined(tag).then(() => resolve());
        } else if (canOptimizeUsingProxy) {
            if (resolvers.has(tag)) {
                resolvers.get(tag).push(resolve);
            } else {
                resolvers.set(tag, [resolve]);
            }
```

```
            document.dispatchEvent(
                new CustomEvent("__darkreader__addUndefinedResolver", {
                    detail: {tag}
                })
            );
        } else {
            const checkIfDefined = () => {
                const elements = undefinedGroups.get(tag);
                if (elements && elements.size > 0) {
                    if (
                        elements.values().next().value.matches(":defined")
                    ) {
                        resolve();
                    } else {
                        requestAnimationFrame(checkIfDefined);
                    }
                }
            };
            requestAnimationFrame(checkIfDefined);
        }
    });
}
function watchWhenCustomElementsDefined(callback) {
    elementsDefinitionCallback = callback;
}
function unsubscribeFromDefineCustomElements() {
    elementsDefinitionCallback = null;
    undefinedGroups.clear();
    document.removeEventListener(
        "__darkreader__isDefined",
        handleIsDefined
    );
}
function watchForStyleChanges(currentStyles, update, shadowRootDiscovered) {
    stopWatchingForStyleChanges();
    const prevStyles = new Set(currentStyles);
    const prevStyleSiblings = new WeakMap();
    const nextStyleSiblings = new WeakMap();
    function saveStylePosition(style) {
        prevStyleSiblings.set(style, style.previousElementSibling);
        nextStyleSiblings.set(style, style.nextElementSibling);
    }
    function forgetStylePosition(style) {
        prevStyleSiblings.delete(style);
        nextStyleSiblings.delete(style);
    }
    function didStylePositionChange(style) {
        return (
            style.previousElementSibling !== prevStyleSiblings.get(style) ||
            style.nextElementSibling !== nextStyleSiblings.get(style)
        );
    }
    currentStyles.forEach(saveStylePosition);
    function handleStyleOperations(operations) {
        const {createdStyles, removedStyles, movedStyles} = operations;
        createdStyles.forEach((s) => saveStylePosition(s));
        movedStyles.forEach((s) => saveStylePosition(s));
        removedStyles.forEach((s) => forgetStylePosition(s));
        createdStyles.forEach((s) => prevStyles.add(s));
        removedStyles.forEach((s) => prevStyles.delete(s));
        if (
            createdStyles.size + removedStyles.size + movedStyles.size >
            0
        ) {
            update({
                created: Array.from(createdStyles),
                removed: Array.from(removedStyles),
                moved: Array.from(movedStyles),
                updated: []
```

```
        });
    }
}
function handleMinorTreeMutations({additions, moves, deletions}) {
    const createdStyles = new Set();
    const removedStyles = new Set();
    const movedStyles = new Set();
    additions.forEach((node) =>
        getManageableStyles(node).forEach((style) =>
            createdStyles.add(style)
        )
    );
    deletions.forEach((node) =>
        getManageableStyles(node).forEach((style) =>
            removedStyles.add(style)
        )
    );
    moves.forEach((node) =>
        getManageableStyles(node).forEach((style) =>
            movedStyles.add(style)
        )
    );
    handleStyleOperations({createdStyles, removedStyles, movedStyles});
    additions.forEach((n) => {
        extendedIterateShadowHosts(n);
        collectUndefinedElements(n);
    });
    additions.forEach(
        (node) => isCustomElement(node) && recordUndefinedElement(node)
    );
}
function handleHugeTreeMutations(root) {
    const styles = new Set(getManageableStyles(root));
    const createdStyles = new Set();
    const removedStyles = new Set();
    const movedStyles = new Set();
    styles.forEach((s) => {
        if (!prevStyles.has(s)) {
            createdStyles.add(s);
        }
    });
    prevStyles.forEach((s) => {
        if (!styles.has(s)) {
            removedStyles.add(s);
        }
    });
    styles.forEach((s) => {
        if (
            !createdStyles.has(s) &&
            !removedStyles.has(s) &&
            didStylePositionChange(s)
        ) {
            movedStyles.add(s);
        }
    });
    handleStyleOperations({createdStyles, removedStyles, movedStyles});
    extendedIterateShadowHosts(root);
    collectUndefinedElements(root);
}
function handleAttributeMutations(mutations) {
    const updatedStyles = new Set();
    const removedStyles = new Set();
    mutations.forEach((m) => {
        const {target} = m;
        if (target.isConnected) {
            if (shouldManageStyle(target)) {
                updatedStyles.add(target);
            } else if (
                target instanceof HTMLLinkElement &&
```

```
                    target.disabled
                ) {
                    removedStyles.add(target);
                }
            }
        });
        if (updatedStyles.size + removedStyles.size > 0) {
            update({
                updated: Array.from(updatedStyles),
                created: [],
                removed: Array.from(removedStyles),
                moved: []
            });
        }
    }
    function observe(root) {
        if (observedRoots.has(root)) {
            return;
        }
        const treeObserver = createOptimizedTreeObserver(root, {
            onMinorMutations: handleMinorTreeMutations,
            onHugeMutations: handleHugeTreeMutations
        });
        const attrObserver = new MutationObserver(handleAttributeMutations);
        attrObserver.observe(root, {
            attributeFilter: ["rel", "disabled", "media", "href"],
            subtree: true
        });
        observers.push(treeObserver, attrObserver);
        observedRoots.add(root);
    }
    function subscribeForShadowRootChanges(node) {
        const {shadowRoot} = node;
        if (shadowRoot == null || observedRoots.has(shadowRoot)) {
            return;
        }
        observe(shadowRoot);
        shadowRootDiscovered(shadowRoot);
    }
    function extendedIterateShadowHosts(node) {
        iterateShadowHosts(node, subscribeForShadowRootChanges);
    }
    observe(document);
    extendedIterateShadowHosts(document.documentElement);
    watchWhenCustomElementsDefined((hosts) => {
        const newStyles = [];
        hosts.forEach((host) =>
            push(newStyles, getManageableStyles(host.shadowRoot))
        );
        update({created: newStyles, updated: [], removed: [], moved: []});
        hosts.forEach((host) => {
            const {shadowRoot} = host;
            if (shadowRoot == null) {
                return;
            }
            subscribeForShadowRootChanges(host);
            extendedIterateShadowHosts(shadowRoot);
            collectUndefinedElements(shadowRoot);
        });
    });
    document.addEventListener("__darkreader__isDefined", handleIsDefined, {
        passive: true
    });
    collectUndefinedElements(document);
}
function resetObservers() {
    observers.forEach((o) => o.disconnect());
    observers.splice(0, observers.length);
    observedRoots = new WeakSet();
```

```
}
function stopWatchingForStyleChanges() {
    resetObservers();
    unsubscribeFromDefineCustomElements();
}

const overrides = new WeakSet();
function hasAdoptedStyleSheets(node) {
    return (
        Array.isArray(node.adoptedStyleSheets) &&
        node.adoptedStyleSheets.length > 0
    );
}
function createAdoptedStyleSheetOverride(node) {
    let cancelAsyncOperations = false;
    function iterateSourceSheets(iterator) {
        node.adoptedStyleSheets.forEach((sheet) => {
            if (!overrides.has(sheet)) {
                iterator(sheet);
            }
        });
    }
    function injectSheet(sheet, override) {
        const newSheets = [...node.adoptedStyleSheets];
        const sheetIndex = newSheets.indexOf(sheet);
        const overrideIndex = newSheets.indexOf(override);
        if (overrideIndex >= 0) {
            newSheets.splice(overrideIndex, 1);
        }
        newSheets.splice(sheetIndex + 1, 0, override);
        node.adoptedStyleSheets = newSheets;
    }
    function clear() {
        const newSheets = [...node.adoptedStyleSheets];
        for (let i = newSheets.length - 1; i >= 0; i--) {
            const sheet = newSheets[i];
            if (overrides.has(sheet)) {
                newSheets.splice(i, 1);
                overrides.delete(sheet);
            }
        }
        if (node.adoptedStyleSheets.length !== newSheets.length) {
            node.adoptedStyleSheets = newSheets;
        }
    }
    function destroy() {
        cancelAsyncOperations = true;
        clear();
        if (frameId) {
            cancelAnimationFrame(frameId);
            frameId = null;
        }
    }
    let rulesChangeKey = 0;
    function getRulesChangeKey() {
        let count = 0;
        iterateSourceSheets((sheet) => {
            count += sheet.cssRules.length;
        });
        if (count === 1) {
            const rule = node.adoptedStyleSheets[0].cssRules[0];
            return rule instanceof CSSStyleRule ? rule.style.length : count;
        }
        return count;
    }
    function render(theme, ignoreImageAnalysis) {
        clear();
        for (let i = node.adoptedStyleSheets.length - 1; i >= 0; i--) {
            const sheet = node.adoptedStyleSheets[i];
```

```
            if (overrides.has(sheet)) {
                continue;
            }
            const rules = sheet.cssRules;
            const override = new CSSStyleSheet();
            const prepareSheet = () => {
                for (let i = override.cssRules.length - 1; i >= 0; i--) {
                    override.deleteRule(i);
                }
                injectSheet(sheet, override);
                overrides.add(override);
                return override;
            };
            const sheetModifier = createStyleSheetModifier();
            sheetModifier.modifySheet({
                prepareSheet,
                sourceCSSRules: rules,
                theme,
                ignoreImageAnalysis,
                force: false,
                isAsyncCancelled: () => cancelAsyncOperations
            });
        }
        rulesChangeKey = getRulesChangeKey();
    }
    function checkForUpdates() {
        return getRulesChangeKey() !== rulesChangeKey;
    }
    let frameId = null;
    function watch(callback) {
        frameId = requestAnimationFrame(() => {
            if (checkForUpdates()) {
                const sheets = node.adoptedStyleSheets.filter(
                    (s) => !overrides.has(s)
                );
                callback(sheets);
            }
            watch(callback);
        });
    }
    return {
        render,
        destroy,
        watch
    };
}

function injectProxy(
    enableStyleSheetsProxy,
    enableCustomElementRegistryProxy
) {
    document.dispatchEvent(
        new CustomEvent("__darkreader__inlineScriptsAllowed")
    );
    const addRuleDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "addRule"
    );
    const insertRuleDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "insertRule"
    );
    const deleteRuleDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "deleteRule"
    );
    const removeRuleDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "removeRule"
```

```
);
const replaceDescriptor = Object.getOwnPropertyDescriptor(
    CSSStyleSheet.prototype,
    "replace"
);
const replaceSyncDescriptor = Object.getOwnPropertyDescriptor(
    CSSStyleSheet.prototype,
    "replaceSync"
);
const documentStyleSheetsDescriptor = enableStyleSheetsProxy
    ? Object.getOwnPropertyDescriptor(Document.prototype, "styleSheets")
    : null;
const customElementRegistryDefineDescriptor =
    enableCustomElementRegistryProxy
        ? Object.getOwnPropertyDescriptor(
              CustomElementRegistry.prototype,
              "define"
          )
        : null;
const shouldWrapHTMLElement = [
    "baidu.com",
    "baike.baidu.com",
    "ditu.baidu.com",
    "map.baidu.com",
    "maps.baidu.com",
    "haokan.baidu.com",
    "pan.baidu.com",
    "passport.baidu.com",
    "tieba.baidu.com",
    "www.baidu.com"
].includes(location.hostname);
const getElementsByTagNameDescriptor = shouldWrapHTMLElement
    ? Object.getOwnPropertyDescriptor(
          Element.prototype,
          "getElementsByTagName"
      )
    : null;
const shouldProxyChildNodes = location.hostname === "www.vy.no";
const childNodesDescriptor = shouldProxyChildNodes
    ? Object.getOwnPropertyDescriptor(Node.prototype, "childNodes")
    : null;
const cleaners = [];
const cleanUp = () => {
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "addRule",
        addRuleDescriptor
    );
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "insertRule",
        insertRuleDescriptor
    );
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "deleteRule",
        deleteRuleDescriptor
    );
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "removeRule",
        removeRuleDescriptor
    );
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "replace",
        replaceDescriptor
    );
    Object.defineProperty(
```

```
        CSSStyleSheet.prototype,
        "replaceSync",
        replaceSyncDescriptor
    );
    document.removeEventListener("__darkreader__cleanUp", cleanUp);
    document.removeEventListener(
        "__darkreader__addUndefinedResolver",
        addUndefinedResolver
    );
    document.removeEventListener(
        "__darkreader__blobURLCheckRequest",
        checkBlobURLSupport
    );
    if (enableStyleSheetsProxy) {
        Object.defineProperty(
            Document.prototype,
            "styleSheets",
            documentStyleSheetsDescriptor
        );
    }
    if (enableCustomElementRegistryProxy) {
        Object.defineProperty(
            CustomElementRegistry.prototype,
            "define",
            customElementRegistryDefineDescriptor
        );
    }
    if (shouldWrapHTMLElement) {
        Object.defineProperty(
            Element.prototype,
            "getElementsByTagName",
            getElementsByTagNameDescriptor
        );
    }
    if (shouldProxyChildNodes) {
        Object.defineProperty(
            Node.prototype,
            "childNodes",
            childNodesDescriptor
        );
    }
    cleaners.forEach((clean) => clean());
    cleaners.splice(0);
};
const addUndefinedResolverInner = (tag) => {
    customElements.whenDefined(tag).then(() => {
        document.dispatchEvent(
            new CustomEvent("__darkreader__isDefined", {detail: {tag}})
        );
    });
};
const addUndefinedResolver = (e) =>
    addUndefinedResolverInner(e.detail.tag);
document.addEventListener("__darkreader__cleanUp", cleanUp, {
    passive: true
});
document.addEventListener(
    "__darkreader__addUndefinedResolver",
    addUndefinedResolver,
    {passive: true}
);
document.addEventListener(
    "__darkreader__blobURLCheckRequest",
    checkBlobURLSupport,
    {once: true}
);
const updateSheetEvent = new Event("__darkreader__updateSheet");
function proxyAddRule(selector, style, index) {
    addRuleDescriptor.value.call(this, selector, style, index);
```

```
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
    return -1;
}
function proxyInsertRule(rule, index) {
    const returnValue = insertRuleDescriptor.value.call(
        this,
        rule,
        index
    );
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
    return returnValue;
}
function proxyDeleteRule(index) {
    deleteRuleDescriptor.value.call(this, index);
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
}
function proxyRemoveRule(index) {
    removeRuleDescriptor.value.call(this, index);
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
}
function proxyReplace(cssText) {
    const returnValue = replaceDescriptor.value.call(this, cssText);
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        ) &&
        returnValue &&
        returnValue instanceof Promise
    ) {
        returnValue.then(() =>
            this.ownerNode.dispatchEvent(updateSheetEvent)
        );
    }
    return returnValue;
}
```

```
function proxyReplaceSync(cssText) {
    replaceSyncDescriptor.value.call(this, cssText);
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
}
function proxyDocumentStyleSheets() {
    const getCurrentValue = () => {
        const docSheets = documentStyleSheetsDescriptor.get.call(this);
        const filteredSheets = [...docSheets].filter(
            (styleSheet) =>
                styleSheet.ownerNode &&
                !(
                    styleSheet.ownerNode.classList &&
                    styleSheet.ownerNode.classList.contains(
                        "darkreader"
                    )
                )
        );
        filteredSheets.item = (item) => filteredSheets[item];
        return Object.setPrototypeOf(
            filteredSheets,
            StyleSheetList.prototype
        );
    };
    let elements = getCurrentValue();
    const styleSheetListBehavior = {
        get: function (_, property) {
            return getCurrentValue()[property];
        }
    };
    elements = new Proxy(elements, styleSheetListBehavior);
    return elements;
}
function proxyCustomElementRegistryDefine(name, constructor, options) {
    addUndefinedResolverInner(name);
    customElementRegistryDefineDescriptor.value.call(
        this,
        name,
        constructor,
        options
    );
}
function proxyGetElementsByTagName(tagName) {
    if (tagName !== "style") {
        return getElementsByTagNameDescriptor.value.call(this, tagName);
    }
    const getCurrentElementValue = () => {
        const elements = getElementsByTagNameDescriptor.value.call(
            this,
            tagName
        );
        return Object.setPrototypeOf(
            [...elements].filter(
                (element) =>
                    element &&
                    !(
                        element.classList &&
                        element.classList.contains("darkreader")
                    )
            ),
            NodeList.prototype
        );
```

```
            };
            let elements = getCurrentElementValue();
            const nodeListBehavior = {
                get: function (_, property) {
                    return getCurrentElementValue()[
                        Number(property) || property
                    ];
                }
            };
            elements = new Proxy(elements, nodeListBehavior);
            return elements;
        }
        function proxyChildNodes() {
            const childNodes = childNodesDescriptor.get.call(this);
            return Object.setPrototypeOf(
                [...childNodes].filter((element) => {
                    return (
                        !element.classList ||
                        !element.classList.contains("darkreader")
                    );
                }),
                NodeList.prototype
            );
        }
        async function checkBlobURLSupport() {
            const svg =
                '<svg xmlns="http://www.w3.org/2000/svg" width="1" height="1"><rect width="1" height="1"
fill="transparent"/></svg>';
            const bytes = new Uint8Array(svg.length);
            for (let i = 0; i < svg.length; i++) {
                bytes[i] = svg.charCodeAt(i);
            }
            const blob = new Blob([bytes], {type: "image/svg+xml"});
            const objectURL = URL.createObjectURL(blob);
            let blobURLAllowed;
            try {
                const image = new Image();
                await new Promise((resolve, reject) => {
                    image.onload = () => resolve();
                    image.onerror = () => reject();
                    image.src = objectURL;
                });
                blobURLAllowed = true;
            } catch (err) {
                blobURLAllowed = false;
            }
            document.dispatchEvent(
                new CustomEvent("__darkreader__blobURLCheckResponse", {
                    detail: {blobURLAllowed}
                })
            );
        }
        Object.defineProperty(CSSStyleSheet.prototype, "addRule", {
            ...addRuleDescriptor,
            value: proxyAddRule
        });
        Object.defineProperty(CSSStyleSheet.prototype, "insertRule", {
            ...insertRuleDescriptor,
            value: proxyInsertRule
        });
        Object.defineProperty(CSSStyleSheet.prototype, "deleteRule", {
            ...deleteRuleDescriptor,
            value: proxyDeleteRule
        });
        Object.defineProperty(CSSStyleSheet.prototype, "removeRule", {
            ...removeRuleDescriptor,
            value: proxyRemoveRule
        });
        Object.defineProperty(CSSStyleSheet.prototype, "replace", {
```

```
        ...replaceDescriptor,
        value: proxyReplace
    });
    Object.defineProperty(CSSStyleSheet.prototype, "replaceSync", {
        ...replaceSyncDescriptor,
        value: proxyReplaceSync
    });
    if (enableStyleSheetsProxy) {
        Object.defineProperty(Document.prototype, "styleSheets", {
            ...documentStyleSheetsDescriptor,
            get: proxyDocumentStyleSheets
        });
    }
    if (enableCustomElementRegistryProxy) {
        Object.defineProperty(CustomElementRegistry.prototype, "define", {
            ...customElementRegistryDefineDescriptor,
            value: proxyCustomElementRegistryDefine
        });
    }
    if (shouldWrapHTMLElement) {
        Object.defineProperty(Element.prototype, "getElementsByTagName", {
            ...getElementsByTagNameDescriptor,
            value: proxyGetElementsByTagName
        });
    }
    if (shouldProxyChildNodes) {
        Object.defineProperty(Node.prototype, "childNodes", {
            ...childNodesDescriptor,
            get: proxyChildNodes
        });
    }
}

let documentVisibilityListener = null;
let documentIsVisible_ = !document.hidden;
const listenerOptions = {
    capture: true,
    passive: true
};
function watchForDocumentVisibility() {
    document.addEventListener(
        "visibilitychange",
        documentVisibilityListener,
        listenerOptions
    );
    window.addEventListener(
        "pageshow",
        documentVisibilityListener,
        listenerOptions
    );
    window.addEventListener(
        "focus",
        documentVisibilityListener,
        listenerOptions
    );
}
function stopWatchingForDocumentVisibility() {
    document.removeEventListener(
        "visibilitychange",
        documentVisibilityListener,
        listenerOptions
    );
    window.removeEventListener(
        "pageshow",
        documentVisibilityListener,
        listenerOptions
    );
    window.removeEventListener(
        "focus",
```

```
        documentVisibilityListener,
        listenerOptions
    );
}
function setDocumentVisibilityListener(callback) {
    const alreadyWatching = Boolean(documentVisibilityListener);
    documentVisibilityListener = () => {
        if (!document.hidden) {
            removeDocumentVisibilityListener();
            callback();
            documentIsVisible_ = true;
        }
    };
    if (!alreadyWatching) {
        watchForDocumentVisibility();
    }
}
function removeDocumentVisibilityListener() {
    stopWatchingForDocumentVisibility();
    documentVisibilityListener = null;
}
function documentIsVisible() {
    return documentIsVisible_;
}

function findRelevantFix(documentURL, fixes) {
    if (
        !Array.isArray(fixes) ||
        fixes.length === 0 ||
        fixes[0].url[0] !== "*"
    ) {
        return null;
    }
    let maxSpecificity = 0;
    let maxSpecificityIndex = null;
    for (let i = 1; i < fixes.length; i++) {
        if (isURLInList(documentURL, fixes[i].url)) {
            const specificity = fixes[i].url[0].length;
            if (
                maxSpecificityIndex === null ||
                maxSpecificity < specificity
            ) {
                maxSpecificity = specificity;
                maxSpecificityIndex = i;
            }
        }
    }
    return maxSpecificityIndex;
}
function combineFixes(fixes) {
    if (fixes.length === 0 || fixes[0].url[0] !== "*") {
        return null;
    }
    function combineArrays(arrays) {
        return arrays.filter(Boolean).flat();
    }
    return {
        url: [],
        invert: combineArrays(fixes.map((fix) => fix.invert)),
        css: fixes
            .map((fix) => fix.css)
            .filter(Boolean)
            .join("\n"),
        ignoreInlineStyle: combineArrays(
            fixes.map((fix) => fix.ignoreInlineStyle)
        ),
        ignoreImageAnalysis: combineArrays(
            fixes.map((fix) => fix.ignoreImageAnalysis)
        ),
```

```
        disableStyleSheetsProxy: fixes.some(
            (fix) => fix.disableStyleSheetsProxy
        ),
        disableCustomElementRegistryProxy: fixes.some(
            (fix) => fix.disableCustomElementRegistryProxy
        )
    };
}

const INSTANCE_ID = generateUID();
const styleManagers = new Map();
const adoptedStyleManagers = [];
const adoptedStyleFallbacks = new Map();
let filter = null;
let fixes = null;
let isIFrame = null;
let ignoredImageAnalysisSelectors = [];
let ignoredInlineSelectors = [];
function createOrUpdateStyle(className, root = document.head || document) {
    let element = root.querySelector(`.${className}`);
    if (!element) {
        element = document.createElement("style");
        element.classList.add("darkreader");
        element.classList.add(className);
        element.media = "screen";
        element.textContent = "";
    }
    return element;
}
function createOrUpdateScript(className, root = document.head || document) {
    let element = root.querySelector(`.${className}`);
    if (!element) {
        element = document.createElement("script");
        element.classList.add("darkreader");
        element.classList.add(className);
    }
    return element;
}
const nodePositionWatchers = new Map();
function setupNodePositionWatcher(node, alias) {
    nodePositionWatchers.has(alias) &&
        nodePositionWatchers.get(alias).stop();
    nodePositionWatchers.set(alias, watchForNodePosition(node, "head"));
}
function stopStylePositionWatchers() {
    forEach(nodePositionWatchers.values(), (watcher) => watcher.stop());
    nodePositionWatchers.clear();
}
function createStaticStyleOverrides() {
    const fallbackStyle = createOrUpdateStyle(
        "darkreader--fallback",
        document
    );
    fallbackStyle.textContent = getModifiedFallbackStyle(filter, {
        strict: true
    });
    document.head.insertBefore(fallbackStyle, document.head.firstChild);
    setupNodePositionWatcher(fallbackStyle, "fallback");
    const userAgentStyle = createOrUpdateStyle("darkreader--user-agent");
    userAgentStyle.textContent = getModifiedUserAgentStyle(
        filter,
        isIFrame,
        filter.styleSystemControls
    );
    document.head.insertBefore(userAgentStyle, fallbackStyle.nextSibling);
    setupNodePositionWatcher(userAgentStyle, "user-agent");
    const textStyle = createOrUpdateStyle("darkreader--text");
    if (filter.useFont || filter.textStroke > 0) {
        textStyle.textContent = createTextStyle(filter);
```

```
} else {
    textStyle.textContent = "";
}
document.head.insertBefore(textStyle, fallbackStyle.nextSibling);
setupNodePositionWatcher(textStyle, "text");
const invertStyle = createOrUpdateStyle("darkreader--invert");
if (fixes && Array.isArray(fixes.invert) && fixes.invert.length > 0) {
    invertStyle.textContent = [
        `${fixes.invert.join(", ")} {`,
        `    filter: ${getCSSFilterValue({
            ...filter,
            contrast:
                filter.mode === 0
                    ? filter.contrast
                    : clamp(filter.contrast - 10, 0, 100)
        })} !important;`,
        "}"
    ].join("\n");
} else {
    invertStyle.textContent = "";
}
document.head.insertBefore(invertStyle, textStyle.nextSibling);
setupNodePositionWatcher(invertStyle, "invert");
const inlineStyle = createOrUpdateStyle("darkreader--inline");
inlineStyle.textContent = getInlineOverrideStyle();
document.head.insertBefore(inlineStyle, invertStyle.nextSibling);
setupNodePositionWatcher(inlineStyle, "inline");
const overrideStyle = createOrUpdateStyle("darkreader--override");
overrideStyle.textContent =
    fixes && fixes.css ? replaceCSSTemplates(fixes.css) : "";
document.head.appendChild(overrideStyle);
setupNodePositionWatcher(overrideStyle, "override");
const variableStyle = createOrUpdateStyle("darkreader--variables");
const selectionColors = getSelectionColor(filter);
const {
    darkSchemeBackgroundColor,
    darkSchemeTextColor,
    lightSchemeBackgroundColor,
    lightSchemeTextColor,
    mode
} = filter;
let schemeBackgroundColor =
    mode === 0 ? lightSchemeBackgroundColor : darkSchemeBackgroundColor;
let schemeTextColor =
    mode === 0 ? lightSchemeTextColor : darkSchemeTextColor;
schemeBackgroundColor = modifyBackgroundColor(
    parseColorWithCache(schemeBackgroundColor),
    filter
);
schemeTextColor = modifyForegroundColor(
    parseColorWithCache(schemeTextColor),
    filter
);
variableStyle.textContent = [
    `:root {`,
    `   --darkreader-neutral-background: ${schemeBackgroundColor};`,
    `   --darkreader-neutral-text: ${schemeTextColor};`,
    `   --darkreader-selection-background: ${selectionColors.backgroundColorSelection};`,
    `   --darkreader-selection-text: ${selectionColors.foregroundColorSelection};`,
    `}`
].join("\n");
document.head.insertBefore(variableStyle, inlineStyle.nextSibling);
setupNodePositionWatcher(variableStyle, "variables");
const rootVarsStyle = createOrUpdateStyle("darkreader--root-vars");
document.head.insertBefore(rootVarsStyle, variableStyle.nextSibling);
const enableStyleSheetsProxy = !(
    fixes && fixes.disableStyleSheetsProxy
);
const enableCustomElementRegistryProxy = !(
```

```
        fixes && fixes.disableCustomElementRegistryProxy
    );
    {
        const proxyScript = createOrUpdateScript("darkreader--proxy");
        proxyScript.append(
            `(${injectProxy})(${enableStyleSheetsProxy}, ${enableCustomElementRegistryProxy})`
        );
        document.head.insertBefore(proxyScript, rootVarsStyle.nextSibling);
        proxyScript.remove();
    }
}
const shadowRootsWithOverrides = new Set();
function createShadowStaticStyleOverridesInner(root) {
    const inlineStyle = createOrUpdateStyle("darkreader--inline", root);
    inlineStyle.textContent = getInlineOverrideStyle();
    root.insertBefore(inlineStyle, root.firstChild);
    const overrideStyle = createOrUpdateStyle("darkreader--override", root);
    overrideStyle.textContent =
        fixes && fixes.css ? replaceCSSTemplates(fixes.css) : "";
    root.insertBefore(overrideStyle, inlineStyle.nextSibling);
    const invertStyle = createOrUpdateStyle("darkreader--invert", root);
    if (fixes && Array.isArray(fixes.invert) && fixes.invert.length > 0) {
        invertStyle.textContent = [
            `${fixes.invert.join(", ")} {`,
            `    filter: ${getCSSFilterValue({
                ...filter,
                contrast:
                    filter.mode === 0
                        ? filter.contrast
                        : clamp(filter.contrast - 10, 0, 100)
            })} !important;`,
            "}"
        ].join("\n");
    } else {
        invertStyle.textContent = "";
    }
    root.insertBefore(invertStyle, overrideStyle.nextSibling);
    shadowRootsWithOverrides.add(root);
}
function delayedCreateShadowStaticStyleOverrides(root) {
    const observer = new MutationObserver((mutations, observer) => {
        observer.disconnect();
        for (const {type, removedNodes} of mutations) {
            if (type === "childList") {
                for (const {nodeName, className} of removedNodes) {
                    if (
                        nodeName === "STYLE" &&
                        [
                            "darkreader darkreader--inline",
                            "darkreader darkreader--override",
                            "darkreader darkreader--invert"
                        ].includes(className)
                    ) {
                        createShadowStaticStyleOverridesInner(root);
                        return;
                    }
                }
            }
        }
    });
    observer.observe(root, {childList: true});
}
function createShadowStaticStyleOverrides(root) {
    const uninit = root.firstChild === null;
    createShadowStaticStyleOverridesInner(root);
    if (uninit) {
        delayedCreateShadowStaticStyleOverrides(root);
    }
}
```

```
function replaceCSSTemplates($cssText) {
    return $cssText.replace(/\${(.+?)}/g, (_, $color) => {
        const color = parseColorWithCache($color);
        if (color) {
            return modifyColor(color, filter);
        }
        return $color;
    });
}
function cleanFallbackStyle() {
    const fallback = document.querySelector(".darkreader--fallback");
    if (fallback) {
        fallback.textContent = "";
    }
}
function createDynamicStyleOverrides() {
    cancelRendering();
    const allStyles = getManageableStyles(document);
    const newManagers = allStyles
        .filter((style) => !styleManagers.has(style))
        .map((style) => createManager(style));
    newManagers
        .map((manager) => manager.details({secondRound: false}))
        .filter((detail) => detail && detail.rules.length > 0)
        .forEach((detail) => {
            variablesStore.addRulesForMatching(detail.rules);
        });
    variablesStore.matchVariablesAndDependents();
    variablesStore.setOnRootVariableChange(() => {
        const rootVarsStyle = createOrUpdateStyle("darkreader--root-vars");
        variablesStore.putRootVars(rootVarsStyle, filter);
    });
    const rootVarsStyle = createOrUpdateStyle("darkreader--root-vars");
    variablesStore.putRootVars(rootVarsStyle, filter);
    styleManagers.forEach((manager) =>
        manager.render(filter, ignoredImageAnalysisSelectors)
    );
    if (loadingStyles.size === 0) {
        cleanFallbackStyle();
    }
    newManagers.forEach((manager) => manager.watch());
    const inlineStyleElements = toArray(
        document.querySelectorAll(INLINE_STYLE_SELECTOR)
    );
    iterateShadowHosts(document.documentElement, (host) => {
        createShadowStaticStyleOverrides(host.shadowRoot);
        const elements = host.shadowRoot.querySelectorAll(
            INLINE_STYLE_SELECTOR
        );
        if (elements.length > 0) {
            push(inlineStyleElements, elements);
        }
    });
    inlineStyleElements.forEach((el) =>
        overrideInlineStyle(
            el,
            filter,
            ignoredInlineSelectors,
            ignoredImageAnalysisSelectors
        )
    );
    handleAdoptedStyleSheets(document);
    variablesStore.matchVariablesAndDependents();
}
let loadingStylesCounter = 0;
const loadingStyles = new Set();
function createManager(element) {
    const loadingStyleId = ++loadingStylesCounter;
    function loadingStart() {
```

```
        if (!isDOMReady() || !documentIsVisible()) {
            loadingStyles.add(loadingStyleId);
            logInfo(
                `Current amount of styles loading: ${loadingStyles.size}`
            );
            const fallbackStyle = document.querySelector(
                ".darkreader--fallback"
            );
            if (!fallbackStyle.textContent) {
                fallbackStyle.textContent = getModifiedFallbackStyle(
                    filter,
                    {strict: false}
                );
            }
        }
    }
    function loadingEnd() {
        loadingStyles.delete(loadingStyleId);
        logInfo(
            `Removed loadingStyle ${loadingStyleId}, now awaiting: ${loadingStyles.size}`
        );
        if (loadingStyles.size === 0 && isDOMReady()) {
            cleanFallbackStyle();
        }
    }
    function update() {
        const details = manager.details({secondRound: true});
        if (!details) {
            return;
        }
        variablesStore.addRulesForMatching(details.rules);
        variablesStore.matchVariablesAndDependents();
        manager.render(filter, ignoredImageAnalysisSelectors);
    }
    const manager = manageStyle(element, {
        update,
        loadingStart,
        loadingEnd
    });
    styleManagers.set(element, manager);
    return manager;
}
function removeManager(element) {
    const manager = styleManagers.get(element);
    if (manager) {
        manager.destroy();
        styleManagers.delete(element);
    }
}
const throttledRenderAllStyles = throttle((callback) => {
    styleManagers.forEach((manager) =>
        manager.render(filter, ignoredImageAnalysisSelectors)
    );
    adoptedStyleManagers.forEach((manager) =>
        manager.render(filter, ignoredImageAnalysisSelectors)
    );
    callback && callback();
});
const cancelRendering = function () {
    throttledRenderAllStyles.cancel();
};
function onDOMReady() {
    if (loadingStyles.size === 0) {
        cleanFallbackStyle();
        return;
    }
}
function runDynamicStyle() {
    createDynamicStyleOverrides();
```

```
    watchForUpdates();
}
function createThemeAndWatchForUpdates() {
    createStaticStyleOverrides();
    if (!documentIsVisible() && !filter.immediateModify) {
        setDocumentVisibilityListener(runDynamicStyle);
    } else {
        runDynamicStyle();
    }
    changeMetaThemeColorWhenAvailable(filter);
}
let pendingAdoptedVarMatch = false;
function handleAdoptedStyleSheets(node) {
    const theme = filter;
    if (hasAdoptedStyleSheets(node)) {
        node.adoptedStyleSheets.forEach((s) => {
            variablesStore.addRulesForMatching(s.cssRules);
        });
        const newManger = createAdoptedStyleSheetOverride(node);
        adoptedStyleManagers.push(newManger);
        newManger.render(theme, ignoredImageAnalysisSelectors);
        newManger.watch((sheets) => {
            sheets.forEach((s) => {
                variablesStore.addRulesForMatching(s.cssRules);
            });
            newManger.render(theme, ignoredImageAnalysisSelectors);
            pendingAdoptedVarMatch = true;
        });
        potentialAdoptedStyleNodes.delete(node);
    } else if (!potentialAdoptedStyleNodes.has(node)) {
        potentialAdoptedStyleNodes.add(node);
    }
}
const potentialAdoptedStyleNodes = new Set();
let potentialAdoptedStyleFrameId = null;
function watchPotentialAdoptedStyleNodes() {
    potentialAdoptedStyleFrameId = requestAnimationFrame(() => {
        let changed = false;
        potentialAdoptedStyleNodes.forEach((node) => {
            if (node.isConnected) {
                handleAdoptedStyleSheets(node);
                changed = true;
            } else {
                potentialAdoptedStyleNodes.delete(node);
            }
        });
        if (changed || pendingAdoptedVarMatch) {
            variablesStore.matchVariablesAndDependents();
            pendingAdoptedVarMatch = false;
        }
        watchPotentialAdoptedStyleNodes();
    });
}
function stopWatchingPotentialAdoptedStyleNodes() {
    potentialAdoptedStyleFrameId &&
        cancelAnimationFrame(potentialAdoptedStyleFrameId);
    potentialAdoptedStyleNodes.clear();
}
function watchForUpdates() {
    const managedStyles = Array.from(styleManagers.keys());
    watchForStyleChanges(
        managedStyles,
        ({created, updated, removed, moved}) => {
            const stylesToRemove = removed;
            const stylesToManage = created
                .concat(updated)
                .concat(moved)
                .filter((style) => !styleManagers.has(style));
            const stylesToRestore = moved.filter((style) =>
```

```
                styleManagers.has(style)
            );
            stylesToRemove.forEach((style) => removeManager(style));
            const newManagers = stylesToManage.map((style) =>
                createManager(style)
            );
            newManagers
                .map((manager) => manager.details({secondRound: false}))
                .filter((detail) => detail && detail.rules.length > 0)
                .forEach((detail) => {
                    variablesStore.addRulesForMatching(detail.rules);
                });
            variablesStore.matchVariablesAndDependents();
            newManagers.forEach((manager) =>
                manager.render(filter, ignoredImageAnalysisSelectors)
            );
            newManagers.forEach((manager) => manager.watch());
            stylesToRestore.forEach((style) =>
                styleManagers.get(style).restore()
            );
        },
        (shadowRoot) => {
            createShadowStaticStyleOverrides(shadowRoot);
            handleAdoptedStyleSheets(shadowRoot);
        }
    );
    watchPotentialAdoptedStyleNodes();
    watchForInlineStyles(
        (element) => {
            overrideInlineStyle(
                element,
                filter,
                ignoredInlineSelectors,
                ignoredImageAnalysisSelectors
            );
            if (element === document.documentElement) {
                const styleAttr = element.getAttribute("style") || "";
                if (styleAttr.includes("--")) {
                    variablesStore.matchVariablesAndDependents();
                    const rootVarsStyle = createOrUpdateStyle(
                        "darkreader--root-vars"
                    );
                    variablesStore.putRootVars(rootVarsStyle, filter);
                }
            }
        },
        (root) => {
            createShadowStaticStyleOverrides(root);
            const inlineStyleElements = root.querySelectorAll(
                INLINE_STYLE_SELECTOR
            );
            if (inlineStyleElements.length > 0) {
                forEach(inlineStyleElements, (el) =>
                    overrideInlineStyle(
                        el,
                        filter,
                        ignoredInlineSelectors,
                        ignoredImageAnalysisSelectors
                    )
                );
            }
        }
    );
    addDOMReadyListener(onDOMReady);
}
function stopWatchingForUpdates() {
    styleManagers.forEach((manager) => manager.pause());
    stopStylePositionWatchers();
    stopWatchingForStyleChanges();
```

```
    stopWatchingForInlineStyles();
    removeDOMReadyListener(onDOMReady);
    cleanReadyStateCompleteListeners();
}
let metaObserver;
function addMetaListener() {
    metaObserver = new MutationObserver(() => {
        if (document.querySelector('meta[name="darkreader-lock"]')) {
            metaObserver.disconnect();
            removeDynamicTheme();
        }
    });
    metaObserver.observe(document.head, {childList: true, subtree: true});
}
function createDarkReaderInstanceMarker() {
    const metaElement = document.createElement("meta");
    metaElement.name = "darkreader";
    metaElement.content = INSTANCE_ID;
    document.head.appendChild(metaElement);
}
function isAnotherDarkReaderInstanceActive() {
    if (document.querySelector('meta[name="darkreader-lock"]')) {
        return true;
    }
    const meta = document.querySelector('meta[name="darkreader"]');
    if (meta) {
        if (meta.content !== INSTANCE_ID) {
            return true;
        }
        return false;
    }
    createDarkReaderInstanceMarker();
    addMetaListener();
    return false;
}
function selectRelevantFix(documentURL, fixes) {
    if (!fixes) {
        return null;
    }
    if (fixes.length === 0 || fixes[0].url[0] !== "*") {
        return null;
    }
    const relevantFixIndex = findRelevantFix(documentURL, fixes);
    return relevantFixIndex
        ? combineFixes([fixes[0], fixes[relevantFixIndex]])
        : fixes[0];
}
function createOrUpdateDynamicTheme(
    filterConfig,
    dynamicThemeFixes,
    iframe
) {
    const dynamicThemeFix = selectRelevantFix(
        document.location.href,
        dynamicThemeFixes
    );
    createOrUpdateDynamicThemeInternal(
        filterConfig,
        dynamicThemeFix,
        iframe
    );
}
function createOrUpdateDynamicThemeInternal(
    filterConfig,
    dynamicThemeFixes,
    iframe
) {
    filter = filterConfig;
    fixes = dynamicThemeFixes;
```

```
    if (fixes) {
        ignoredImageAnalysisSelectors = Array.isArray(
            fixes.ignoreImageAnalysis
        )
            ? fixes.ignoreImageAnalysis
            : [];
        ignoredInlineSelectors = Array.isArray(fixes.ignoreInlineStyle)
            ? fixes.ignoreInlineStyle
            : [];
    } else {
        ignoredImageAnalysisSelectors = [];
        ignoredInlineSelectors = [];
    }
    if (filter.immediateModify) {
        setIsDOMReady(() => {
            return true;
        });
    }
    isIFrame = iframe;
    if (document.head) {
        if (isAnotherDarkReaderInstanceActive()) {
            removeDynamicTheme();
            return;
        }
        document.documentElement.setAttribute(
            "data-darkreader-mode",
            "dynamic"
        );
        document.documentElement.setAttribute(
            "data-darkreader-scheme",
            filter.mode ? "dark" : "dimmed"
        );
        createThemeAndWatchForUpdates();
    } else {
        {
            const fallbackStyle = createOrUpdateStyle(
                "darkreader--fallback"
            );
            document.documentElement.appendChild(fallbackStyle);
            fallbackStyle.textContent = getModifiedFallbackStyle(filter, {
                strict: true
            });
        }
        const headObserver = new MutationObserver(() => {
            if (document.head) {
                headObserver.disconnect();
                if (isAnotherDarkReaderInstanceActive()) {
                    removeDynamicTheme();
                    return;
                }
                createThemeAndWatchForUpdates();
            }
        });
        headObserver.observe(document, {childList: true, subtree: true});
    }
}
function removeProxy() {
    document.dispatchEvent(new CustomEvent("__darkreader__cleanUp"));
    removeNode(document.head.querySelector(".darkreader--proxy"));
}
const cleaners = [];
function removeDynamicTheme() {
    document.documentElement.removeAttribute(`data-darkreader-mode`);
    document.documentElement.removeAttribute(`data-darkreader-scheme`);
    cleanDynamicThemeCache();
    removeNode(document.querySelector(".darkreader--fallback"));
    if (document.head) {
        restoreMetaThemeColor();
        removeNode(document.head.querySelector(".darkreader--user-agent"));
```

```
            removeNode(document.head.querySelector(".darkreader--text"));
            removeNode(document.head.querySelector(".darkreader--invert"));
            removeNode(document.head.querySelector(".darkreader--inline"));
            removeNode(document.head.querySelector(".darkreader--override"));
            removeNode(document.head.querySelector(".darkreader--variables"));
            removeNode(document.head.querySelector(".darkreader--root-vars"));
            removeNode(document.head.querySelector('meta[name="darkreader"]'));
            removeProxy();
        }
        shadowRootsWithOverrides.forEach((root) => {
            removeNode(root.querySelector(".darkreader--inline"));
            removeNode(root.querySelector(".darkreader--override"));
        });
        shadowRootsWithOverrides.clear();
        forEach(styleManagers.keys(), (el) => removeManager(el));
        loadingStyles.clear();
        cleanLoadingLinks();
        forEach(document.querySelectorAll(".darkreader"), removeNode);
        adoptedStyleManagers.forEach((manager) => manager.destroy());
        adoptedStyleManagers.splice(0);
        adoptedStyleFallbacks.forEach((fallback) => fallback.destroy());
        adoptedStyleFallbacks.clear();
        stopWatchingPotentialAdoptedStyleNodes();
        metaObserver && metaObserver.disconnect();
        cleaners.forEach((clean) => clean());
        cleaners.splice(0);
    }
    function cleanDynamicThemeCache() {
        variablesStore.clear();
        parsedURLCache.clear();
        removeDocumentVisibilityListener();
        cancelRendering();
        stopWatchingForUpdates();
        cleanModificationCache();
        clearColorCache();
    }

    const blobRegex = /url\(\"(blob\:.*?)\"\)/g;
    async function replaceBlobs(text) {
        const promises = [];
        getMatches(blobRegex, text, 1).forEach((url) => {
            const promise = loadAsDataURL(url);
            promises.push(promise);
        });
        const data = await Promise.all(promises);
        return text.replace(blobRegex, () => `url("${data.shift()}")`);
    }
    const banner = `/*
                      _______
                     /       \\
                    .==.     .==.
                   ((  ))==((  ))
                  / "=="    "=="\\
                 /____|| || ||___\\
       ________     ____   ________  ___    ___
      |  ___  \\   /    \\  |  ___  \\ |  |  /  /
      |  |  \\  \\ /  /\\  \\  |  |  \\  \\|  |_/  /
      |  |   )  /  /__\\  \\ |  |__/  /|  ___  \\
      |  |__/  /  ______  \\|  ____  \\|  |  \\  \\
_____|_______/__/ ____ \\__\\__|___\\__\\__|___\\__\\____
|  ___  \\ |  ____/ /    \\   |  ___  \\ |  ____|  ___  \\
|  |  \\  \\|  |___ /  /\\  \\  |  |  \\  \\|  |___|  |  \\  \\
|  |__/  /|  ____/  /__\\  \\ |  |   )  |  ____|  |__/  /
|  ____  \\|  |__/  ______  \\|  |__/  /|  |___|  ____  \\
|__|   \\__\\____/__/      \\__\\_______/ |______|__|   \\__\\
                https://darkreader.org
*/

/*! Dark reader generated CSS | Licensed under MIT https://github.com/darkreader/darkreader/blob/main/LICENSE */
```

```
`;
    async function collectCSS() {
        const css = [banner];
        function addStaticCSS(selector, comment) {
            const staticStyle = document.querySelector(selector);
            if (staticStyle && staticStyle.textContent) {
                css.push(`/* ${comment} */`);
                css.push(staticStyle.textContent);
                css.push("");
            }
        }
        addStaticCSS(".darkreader--fallback", "Fallback Style");
        addStaticCSS(".darkreader--user-agent", "User-Agent Style");
        addStaticCSS(".darkreader--text", "Text Style");
        addStaticCSS(".darkreader--invert", "Invert Style");
        addStaticCSS(".darkreader--variables", "Variables Style");
        const modifiedCSS = [];
        document.querySelectorAll(".darkreader--sync").forEach((element) => {
            forEach(element.sheet.cssRules, (rule) => {
                rule && rule.cssText && modifiedCSS.push(rule.cssText);
            });
        });
        if (modifiedCSS.length) {
            const formattedCSS = formatCSS(modifiedCSS.join("\n"));
            css.push("/* Modified CSS */");
            css.push(await replaceBlobs(formattedCSS));
            css.push("");
        }
        addStaticCSS(".darkreader--override", "Override Style");
        return css.join("\n");
    }

    let unloaded = false;
    const scriptId = generateUID();
    function cleanup() {
        unloaded = true;
        removeEventListener("pagehide", onPageHide);
        removeEventListener("freeze", onFreeze);
        removeEventListener("resume", onResume);
        cleanDynamicThemeCache();
        stopDarkThemeDetector();
        stopColorSchemeChangeDetector();
    }
    function sendMessage(message) {
        if (unloaded) {
            return;
        }
        const responseHandler = (response) => {
            if (response === "unsupportedSender") {
                removeStyle();
                removeSVGFilter();
                removeDynamicTheme();
                cleanup();
            }
        };
        try {
            if (false);
            else {
                chrome.runtime.sendMessage(message, responseHandler);
            }
        } catch (error) {
            if (error.message === "Extension context invalidated.") {
                console.log(
                    "Dark Reader: instance of old CS detected, clening up."
                );
                cleanup();
            } else {
                console.log(
                    "Dark Reader: unexpected error during message passing."
```

```
            );
        }
    }
}
function onMessage(message) {
    if (
        message.scriptId !== scriptId &&
        message.type !== MessageTypeUItoCS.EXPORT_CSS
    ) {
        return;
    }
    logInfoCollapsed(`onMessage[${message.type}]`, message);
    switch (message.type) {
        case MessageTypeBGtoCS.ADD_CSS_FILTER:
        case MessageTypeBGtoCS.ADD_STATIC_THEME: {
            const {css, detectDarkTheme, detectorHints} = message.data;
            removeDynamicTheme();
            createOrUpdateStyle$1(
                css,
                message.type === MessageTypeBGtoCS.ADD_STATIC_THEME
                    ? "static"
                    : "filter"
            );
            if (detectDarkTheme) {
                runDarkThemeDetector((hasDarkTheme) => {
                    if (hasDarkTheme) {
                        removeStyle();
                        onDarkThemeDetected();
                    }
                }, detectorHints);
            }
            break;
        }
        case MessageTypeBGtoCS.ADD_SVG_FILTER: {
            const {
                css,
                svgMatrix,
                svgReverseMatrix,
                detectDarkTheme,
                detectorHints
            } = message.data;
            removeDynamicTheme();
            createOrUpdateSVGFilter(svgMatrix, svgReverseMatrix);
            createOrUpdateStyle$1(css, "filter");
            if (detectDarkTheme) {
                runDarkThemeDetector((hasDarkTheme) => {
                    if (hasDarkTheme) {
                        removeStyle();
                        removeSVGFilter();
                        onDarkThemeDetected();
                    }
                }, detectorHints);
            }
            break;
        }
        case MessageTypeBGtoCS.ADD_DYNAMIC_THEME: {
            const {theme, fixes, isIFrame, detectDarkTheme, detectorHints} =
                message.data;
            removeStyle();
            createOrUpdateDynamicTheme(theme, fixes, isIFrame);
            if (detectDarkTheme) {
                runDarkThemeDetector((hasDarkTheme) => {
                    if (hasDarkTheme) {
                        removeDynamicTheme();
                        onDarkThemeDetected();
                    }
                }, detectorHints);
            }
            break;
```

```
            }
            case MessageTypeUItoCS.EXPORT_CSS:
                collectCSS().then((collectedCSS) =>
                    sendMessage({
                        type: MessageTypeCStoUI.EXPORT_CSS_RESPONSE,
                        data: collectedCSS
                    })
                );
                break;
            case MessageTypeBGtoCS.UNSUPPORTED_SENDER:
            case MessageTypeBGtoCS.CLEAN_UP:
                removeStyle();
                removeSVGFilter();
                removeDynamicTheme();
                stopDarkThemeDetector();
                break;
        }
    }
    function sendConnectionOrResumeMessage(type) {
        sendMessage({
            type,
            scriptId,
            data: {
                isDark: isSystemDarkModeEnabled(),
                isTopFrame: window === window.top
            }
        });
    }
    runColorSchemeChangeDetector((isDark) =>
        sendMessage({
            type: MessageTypeCStoBG.COLOR_SCHEME_CHANGE,
            data: {isDark}
        })
    );
    chrome.runtime.onMessage.addListener(onMessage);
    sendConnectionOrResumeMessage(MessageTypeCStoBG.DOCUMENT_CONNECT);
    function onPageHide(e) {
        if (e.persisted === false) {
            sendMessage({type: MessageTypeCStoBG.DOCUMENT_FORGET, scriptId});
        }
    }
    function onFreeze() {
        sendMessage({type: MessageTypeCStoBG.DOCUMENT_FREEZE});
    }
    function onResume() {
        sendConnectionOrResumeMessage(MessageTypeCStoBG.DOCUMENT_RESUME);
    }
    function onDarkThemeDetected() {
        sendMessage({type: MessageTypeCStoBG.DARK_THEME_DETECTED});
    }
    {
        addEventListener("pagehide", onPageHide, {passive: true});
        addEventListener("freeze", onFreeze, {passive: true});
        addEventListener("resume", onResume, {passive: true});
    }
})();(self.webpackChunk_N_E = self.webpackChunk_N_E || []).push([[774], {
    2703: function(e, t, n) {
        "use strict";
        var r = n(414);
        function l() {}
        function i() {}
        i.resetWarningCache = l,
        e.exports = function() {
            function e(e, t, n, l, i, a) {
                if (a !== r) {
                    var o = new Error("Calling PropTypes validators directly is not supported by the `prop-types`
package. Use PropTypes.checkPropTypes() to call them. Read more at http://fb.me/use-check-prop-types");
                    throw o.name = "Invariant Violation",
                    o
```

```
            }
        }
        function t() {
            return e
        }
        e.isRequired = e;
        var n = {
            array: e,
            bigint: e,
            bool: e,
            func: e,
            number: e,
            object: e,
            string: e,
            symbol: e,
            any: e,
            arrayOf: t,
            element: e,
            elementType: e,
            instanceOf: t,
            node: e,
            objectOf: t,
            oneOf: t,
            oneOfType: t,
            shape: t,
            exact: t,
            checkPropTypes: i,
            resetWarningCache: l
        };
        return n.PropTypes = n,
        n
    }
},
5697: function(e, t, n) {
    e.exports = n(2703)()
},
414: function(e) {
    "use strict";
    e.exports = "SECRET_DO_NOT_PASS_THIS_OR_YOU_WILL_BE_FIRED"
},
4448: function(e, t, n) {
    "use strict";
    var r = n(3027)
      , l = n(6086)
      , i = n(4142);
    function a(e) {
        for (var t = "https://reactjs.org/docs/error-decoder.html?invariant=" + e, n = 1; n < arguments.length;
n++)
            t += "&args[]=" + encodeURIComponent(arguments[n]);
        return "Minified React error #" + e + "; visit " + t + " for the full message or use the non-minified
dev environment for full errors and additional helpful warnings."
    }
    if (!r)
        throw Error(a(227));
    function o(e, t, n, r, l, i, a, o, u) {
        var c = Array.prototype.slice.call(arguments, 3);
        try {
            t.apply(n, c)
        } catch (s) {
            this.onError(s)
        }
    }
    var u = !1
      , c = null
      , s = !1
      , f = null
      , d = {
        onError: function(e) {
            u = !0,
```

```
        c = e
    }
};
function p(e, t, n, r, l, i, a, s, f) {
    u = !1,
    c = null,
    o.apply(d, arguments)
}
var m = null
  , h = null
  , v = null;
function y(e, t, n) {
    var r = e.type || "unknown-event";
    e.currentTarget = v(n),
    function(e, t, n, r, l, i, o, d, m) {
        if (p.apply(this, arguments),
        u) {
            if (!u)
                throw Error(a(198));
            var h = c;
            u = !1,
            c = null,
            s || (s = !0,
            f = h)
        }
    }(r, t, void 0, e),
    e.currentTarget = null
}
var g = null
  , b = {};
function w() {
    if (g)
        for (var e in b) {
            var t = b[e]
              , n = g.indexOf(e);
            if (!(-1 < n))
                throw Error(a(96, e));
            if (!x[n]) {
                if (!t.extractEvents)
                    throw Error(a(97, e));
                for (var r in x[n] = t,
                n = t.eventTypes) {
                    var l = void 0
                      , i = n[r]
                      , o = t
                      , u = r;
                    if (T.hasOwnProperty(u))
                        throw Error(a(99, u));
                    T[u] = i;
                    var c = i.phasedRegistrationNames;
                    if (c) {
                        for (l in c)
                            c.hasOwnProperty(l) && k(c[l], o, u);
                        l = !0
                    } else
                        i.registrationName ? (k(i.registrationName, o, u),
                        l = !0) : l = !1;
                    if (!l)
                        throw Error(a(98, r, e))
                }
            }
        }
}
function k(e, t, n) {
    if (E[e])
        throw Error(a(100, e));
    E[e] = t,
    S[e] = t.eventTypes[n].dependencies
}
```

```
        var x = []
          , T = {}
          , E = {}
          , S = {};
        function C(e) {
            var t, n = !1;
            for (t in e)
                if (e.hasOwnProperty(t)) {
                    var r = e[t];
                    if (!b.hasOwnProperty(t) || b[t] !== r) {
                        if (b[t])
                            throw Error(a(102, t));
                        b[t] = r,
                        n = !0
                    }
                }
            n && w()
        }
        var _ = !("undefined" === typeof window || "undefined" === typeof window.document || "undefined" === typeof
window.document.createElement)
          , P = null
          , N = null
          , z = null;
        function O(e) {
            if (e = h(e)) {
                if ("function" !== typeof P)
                    throw Error(a(280));
                var t = e.stateNode;
                t && (t = m(t),
                P(e.stateNode, e.type, t))
            }
        }
        function R(e) {
            N ? z ? z.push(e) : z = [e] : N = e
        }
        function I() {
            if (N) {
                var e = N
                  , t = z;
                if (z = N = null,
                O(e),
                t)
                    for (e = 0; e < t.length; e++)
                        O(t[e])
            }
        }
        function M(e, t) {
            return e(t)
        }
        function F(e, t, n, r, l) {
            return e(t, n, r, l)
        }
        function D() {}
        var L = M
          , A = !1
          , U = !1;
        function V() {
            null === N && null === z || (D(),
            I())
        }
        function W(e, t, n) {
            if (U)
                return e(t, n);
            U = !0;
            try {
                return L(e, t, n)
            } finally {
                U = !1,
                V()
```

```
            }
        }
        var j =
/^[:A-Z_a-z\u00C0-\u00D6\u00D8-\u00F6\u00F8-\u02FF\u0370-\u037D\u037F-\u1FFF\u200C-\u200D\u2070-\u218F\u2C00-\u2FEF\
u3001-\uD7FF\uF900-\uFDCF\uFDF0-\uFFFD][:A-Z_a-z\u00C0-\u00D6\u00D8-\u00F6\u00F8-\u02FF\u0370-\u037D\u037F-\u1FFF\u2
00C-\u200D\u2070-\u218F\u2C00-\u2FEF\u3001-\uD7FF\uF900-\uFDCF\uFDF0-\uFFFD\-.0-9\u00B7\u0300-\u036F\u203F-\u2040]*$
/
          , Q = Object.prototype.hasOwnProperty
          , $ = {}
          , B = {};
        function H(e, t, n, r, l, i) {
            this.acceptsBooleans = 2 === t || 3 === t || 4 === t,
            this.attributeName = r,
            this.attributeNamespace = l,
            this.mustUseProperty = n,
            this.propertyName = e,
            this.type = t,
            this.sanitizeURL = i
        }
        var K = {};
        "children dangerouslySetInnerHTML defaultValue defaultChecked innerHTML suppressContentEditableWarning
suppressHydrationWarning style".split(" ").forEach((function(e) {
            K[e] = new H(e,0,!1,e,null,!1)
        }
        )),
        [["acceptCharset", "accept-charset"], ["className", "class"], ["htmlFor", "for"], ["httpEquiv",
"http-equiv"]].forEach((function(e) {
            var t = e[0];
            K[t] = new H(t,1,!1,e[1],null,!1)
        }
        )),
        ["contentEditable", "draggable", "spellCheck", "value"].forEach((function(e) {
            K[e] = new H(e,2,!1,e.toLowerCase(),null,!1)
        }
        )),
        ["autoReverse", "externalResourcesRequired", "focusable", "preserveAlpha"].forEach((function(e) {
            K[e] = new H(e,2,!1,e,null,!1)
        }
        )),
        "allowFullScreen async autoFocus autoPlay controls default defer disabled disablePictureInPicture
formNoValidate hidden loop noModule noValidate open playsInline readOnly required reversed scoped seamless
itemScope".split(" ").forEach((function(e) {
            K[e] = new H(e,3,!1,e.toLowerCase(),null,!1)
        }
        )),
        ["checked", "multiple", "muted", "selected"].forEach((function(e) {
            K[e] = new H(e,3,!0,e,null,!1)
        }
        )),
        ["capture", "download"].forEach((function(e) {
            K[e] = new H(e,4,!1,e,null,!1)
        }
        )),
        ["cols", "rows", "size", "span"].forEach((function(e) {
            K[e] = new H(e,6,!1,e,null,!1)
        }
        )),
        ["rowSpan", "start"].forEach((function(e) {
            K[e] = new H(e,5,!1,e.toLowerCase(),null,!1)
        }
        ));
        var q = /[\-:]([a-z])/g;
        function Y(e) {
            return e[1].toUpperCase()
        }
        "accent-height alignment-baseline arabic-form baseline-shift cap-height clip-path clip-rule
color-interpolation color-interpolation-filters color-profile color-rendering dominant-baseline enable-background
fill-opacity fill-rule flood-color flood-opacity font-family font-size font-size-adjust font-stretch font-style
font-variant font-weight glyph-name glyph-orientation-horizontal glyph-orientation-vertical horiz-adv-x
```

```
horiz-origin-x image-rendering letter-spacing lighting-color marker-end marker-mid marker-start overline-position
overline-thickness paint-order panose-1 pointer-events rendering-intent shape-rendering stop-color stop-opacity
strikethrough-position strikethrough-thickness stroke-dasharray stroke-dashoffset stroke-linecap stroke-linejoin
stroke-miterlimit stroke-opacity stroke-width text-anchor text-decoration text-rendering underline-position
underline-thickness unicode-bidi unicode-range units-per-em v-alphabetic v-hanging v-ideographic v-mathematical
vector-effect vert-adv-y vert-origin-x vert-origin-y word-spacing writing-mode xmlns:xlink x-height".split("
").forEach((function(e) {
            var t = e.replace(q, Y);
            K[t] = new H(t,1,!1,e,null,!1)
        }
        )),
        "xlink:actuate xlink:arcrole xlink:role xlink:show xlink:title xlink:type".split(" ").forEach((function(e) {
            var t = e.replace(q, Y);
            K[t] = new H(t,1,!1,e,"http://www.w3.org/1999/xlink",!1)
        }
        )),
        ["xml:base", "xml:lang", "xml:space"].forEach((function(e) {
            var t = e.replace(q, Y);
            K[t] = new H(t,1,!1,e,"http://www.w3.org/XML/1998/namespace",!1)
        }
        )),
        ["tabIndex", "crossOrigin"].forEach((function(e) {
            K[e] = new H(e,1,!1,e.toLowerCase(),null,!1)
        }
        )),
        K.xlinkHref = new H("xlinkHref",1,!1,"xlink:href","http://www.w3.org/1999/xlink",!0),
        ["src", "href", "action", "formAction"].forEach((function(e) {
            K[e] = new H(e,1,!1,e.toLowerCase(),null,!0)
        }
        ));
        var X = r.__SECRET_INTERNALS_DO_NOT_USE_OR_YOU_WILL_BE_FIRED;
        function G(e, t, n, r) {
            var l = K.hasOwnProperty(t) ? K[t] : null;
            (null !== l ? 0 === l.type : !r && (2 < t.length && ("o" === t[0] || "O" === t[0]) && ("n" === t[1] ||
"N" === t[1]))) || (function(e, t, n, r) {
                if (null === t || "undefined" === typeof t || function(e, t, n, r) {
                    if (null !== n && 0 === n.type)
                        return !1;
                    switch (typeof t) {
                    case "function":
                    case "symbol":
                        return !0;
                    case "boolean":
                        return !r && (null !== n ? !n.acceptsBooleans : "data-" !== (e = e.toLowerCase().slice(0,
5)) && "aria-" !== e);
                    default:
                        return !1
                    }
                }(e, t, n, r))
                    return !0;
                if (r)
                    return !1;
                if (null !== n)
                    switch (n.type) {
                    case 3:
                        return !t;
                    case 4:
                        return !1 === t;
                    case 5:
                        return isNaN(t);
                    case 6:
                        return isNaN(t) || 1 > t
                    }
                return !1
            }(t, n, l, r) && (n = null),
            r || null === l ? function(e) {
                return !!Q.call(B, e) || !Q.call($, e) && (j.test(e) ? B[e] = !0 : ($[e] = !0,
                !1))
            }(t) && (null === n ? e.removeAttribute(t) : e.setAttribute(t, "" + n)) : l.mustUseProperty ?
```

```
e[l.propertyName] = null === n ? 3 !== l.type && "" : n : (t = l.attributeName,
            r = l.attributeNamespace,
            null === n ? e.removeAttribute(t) : (n = 3 === (l = l.type) || 4 === l && !0 === n ? "" : "" + n,
            r ? e.setAttributeNS(r, t, n) : e.setAttribute(t, n))))
        }
        X.hasOwnProperty("ReactCurrentDispatcher") || (X.ReactCurrentDispatcher = {
            current: null
        }),
        X.hasOwnProperty("ReactCurrentBatchConfig") || (X.ReactCurrentBatchConfig = {
            suspense: null
        });
        var Z = /^(.*)[\\\/]/
          , J = "function" === typeof Symbol && Symbol.for
          , ee = J ? Symbol.for("react.element") : 60103
          , te = J ? Symbol.for("react.portal") : 60106
          , ne = J ? Symbol.for("react.fragment") : 60107
          , re = J ? Symbol.for("react.strict_mode") : 60108
          , le = J ? Symbol.for("react.profiler") : 60114
          , ie = J ? Symbol.for("react.provider") : 60109
          , ae = J ? Symbol.for("react.context") : 60110
          , oe = J ? Symbol.for("react.concurrent_mode") : 60111
          , ue = J ? Symbol.for("react.forward_ref") : 60112
          , ce = J ? Symbol.for("react.suspense") : 60113
          , se = J ? Symbol.for("react.suspense_list") : 60120
          , fe = J ? Symbol.for("react.memo") : 60115
          , de = J ? Symbol.for("react.lazy") : 60116
          , pe = J ? Symbol.for("react.block") : 60121
          , me = "function" === typeof Symbol && Symbol.iterator;
        function he(e) {
            return null === e || "object" !== typeof e ? null : "function" === typeof (e = me && e[me] ||
e["@@iterator"]) ? e : null
        }
        function ve(e) {
            if (null == e)
                return null;
            if ("function" === typeof e)
                return e.displayName || e.name || null;
            if ("string" === typeof e)
                return e;
            switch (e) {
            case ne:
                return "Fragment";
            case te:
                return "Portal";
            case le:
                return "Profiler";
            case re:
                return "StrictMode";
            case ce:
                return "Suspense";
            case se:
                return "SuspenseList"
            }
            if ("object" === typeof e)
                switch (e.$$typeof) {
                case ae:
                    return "Context.Consumer";
                case ie:
                    return "Context.Provider";
                case ue:
                    var t = e.render;
                    return t = t.displayName || t.name || "",
                    e.displayName || ("" !== t ? "ForwardRef(" + t + ")" : "ForwardRef");
                case fe:
                    return ve(e.type);
                case pe:
                    return ve(e.render);
                case de:
                    if (e = 1 === e._status ? e._result : null)
```

```
                    return ve(e)
                }
            return null
        }
        function ye(e) {
            var t = "";
            do {
                e: switch (e.tag) {
                case 3:
                case 4:
                case 6:
                case 7:
                case 10:
                case 9:
                    var n = "";
                    break e;
                default:
                    var r = e._debugOwner
                      , l = e._debugSource
                      , i = ve(e.type);
                    n = null,
                    r && (n = ve(r.type)),
                    r = i,
                    i = "",
                    l ? i = " (at " + l.fileName.replace(Z, "") + ":" + l.lineNumber + ")" : n && (i = " (created by
" + n + ")"),
                    n = "\n    in " + (r || "Unknown") + i
                }
                t += n,
                e = e.return
            } while (e);
            return t
        }
        function ge(e) {
            switch (typeof e) {
            case "boolean":
            case "number":
            case "object":
            case "string":
            case "undefined":
                return e;
            default:
                return ""
            }
        }
        function be(e) {
            var t = e.type;
            return (e = e.nodeName) && "input" === e.toLowerCase() && ("checkbox" === t || "radio" === t)
        }
        function we(e) {
            e._valueTracker || (e._valueTracker = function(e) {
                var t = be(e) ? "checked" : "value"
                  , n = Object.getOwnPropertyDescriptor(e.constructor.prototype, t)
                  , r = "" + e[t];
                if (!e.hasOwnProperty(t) && "undefined" !== typeof n && "function" === typeof n.get && "function"
=== typeof n.set) {
                    var l = n.get
                      , i = n.set;
                    return Object.defineProperty(e, t, {
                        configurable: !0,
                        get: function() {
                            return l.call(this)
                        },
                        set: function(e) {
                            r = "" + e,
                            i.call(this, e)
                        }
                    }),
                    Object.defineProperty(e, t, {
```

```
                    enumerable: n.enumerable
                }),
                {
                    getValue: function() {
                        return r
                    },
                    setValue: function(e) {
                        r = "" + e
                    },
                    stopTracking: function() {
                        e._valueTracker = null,
                        delete e[t]
                    }
                }
            }
        }(e))
    }
    function ke(e) {
        if (!e)
            return !1;
        var t = e._valueTracker;
        if (!t)
            return !0;
        var n = t.getValue()
          , r = "";
        return e && (r = be(e) ? e.checked ? "true" : "false" : e.value),
        (e = r) !== n && (t.setValue(e),
        !0)
    }
    function xe(e, t) {
        var n = t.checked;
        return l({}, t, {
            defaultChecked: void 0,
            defaultValue: void 0,
            value: void 0,
            checked: null != n ? n : e._wrapperState.initialChecked
        })
    }
    function Te(e, t) {
        var n = null == t.defaultValue ? "" : t.defaultValue
          , r = null != t.checked ? t.checked : t.defaultChecked;
        n = ge(null != t.value ? t.value : n),
        e._wrapperState = {
            initialChecked: r,
            initialValue: n,
            controlled: "checkbox" === t.type || "radio" === t.type ? null != t.checked : null != t.value
        }
    }
    function Ee(e, t) {
        null != (t = t.checked) && G(e, "checked", t, !1)
    }
    function Se(e, t) {
        Ee(e, t);
        var n = ge(t.value)
          , r = t.type;
        if (null != n)
            "number" === r ? (0 === n && "" === e.value || e.value != n) && (e.value = "" + n) : e.value !== ""
+ n && (e.value = "" + n);
        else if ("submit" === r || "reset" === r)
            return void e.removeAttribute("value");
        t.hasOwnProperty("value") ? _e(e, t.type, n) : t.hasOwnProperty("defaultValue") && _e(e, t.type,
ge(t.defaultValue)),
        null == t.checked && null != t.defaultChecked && (e.defaultChecked = !!t.defaultChecked)
    }
    function Ce(e, t, n) {
        if (t.hasOwnProperty("value") || t.hasOwnProperty("defaultValue")) {
            var r = t.type;
            if (!("submit" !== r && "reset" !== r || void 0 !== t.value && null !== t.value))
                return;
```

```
                t = "" + e._wrapperState.initialValue,
                n || t === e.value || (e.value = t),
                e.defaultValue = t
            }
            "" !== (n = e.name) && (e.name = ""),
            e.defaultChecked = !!e._wrapperState.initialChecked,
            "" !== n && (e.name = n)
        }
        function _e(e, t, n) {
            "number" === t && e.ownerDocument.activeElement === e || (null == n ? e.defaultValue = "" +
e._wrapperState.initialValue : e.defaultValue !== "" + n && (e.defaultValue = "" + n))
        }
        function Pe(e, t) {
            return e = l({
                children: void 0
            }, t),
            (t = function(e) {
                var t = "";
                return r.Children.forEach(e, (function(e) {
                    null != e && (t += e)
                }
                )),
                t
            }(t.children)) && (e.children = t),
            e
        }
        function Ne(e, t, n, r) {
            if (e = e.options,
            t) {
                t = {};
                for (var l = 0; l < n.length; l++)
                    t["$" + n[l]] = !0;
                for (n = 0; n < e.length; n++)
                    l = t.hasOwnProperty("$" + e[n].value),
                    e[n].selected !== l && (e[n].selected = l),
                    l && r && (e[n].defaultSelected = !0)
            } else {
                for (n = "" + ge(n),
                t = null,
                l = 0; l < e.length; l++) {
                    if (e[l].value === n)
                        return e[l].selected = !0,
                        void (r && (e[l].defaultSelected = !0));
                    null !== t || e[l].disabled || (t = e[l])
                }
                null !== t && (t.selected = !0)
            }
        }
        function ze(e, t) {
            if (null != t.dangerouslySetInnerHTML)
                throw Error(a(91));
            return l({}, t, {
                value: void 0,
                defaultValue: void 0,
                children: "" + e._wrapperState.initialValue
            })
        }
        function Oe(e, t) {
            var n = t.value;
            if (null == n) {
                if (n = t.children,
                t = t.defaultValue,
                null != n) {
                    if (null != t)
                        throw Error(a(92));
                    if (Array.isArray(n)) {
                        if (!(1 >= n.length))
                            throw Error(a(93));
                        n = n[0]
```

```
                }
                t = n
            }
            null == t && (t = ""),
            n = t
        }
        e._wrapperState = {
            initialValue: ge(n)
        }
    }
    function Re(e, t) {
        var n = ge(t.value)
          , r = ge(t.defaultValue);
        null != n && ((n = "" + n) !== e.value && (e.value = n),
        null == t.defaultValue && e.defaultValue !== n && (e.defaultValue = n)),
        null != r && (e.defaultValue = "" + r)
    }
    function Ie(e) {
        var t = e.textContent;
        t === e._wrapperState.initialValue && "" !== t && null !== t && (e.value = t)
    }
    var Me = "http://www.w3.org/1999/xhtml"
      , Fe = "http://www.w3.org/2000/svg";
    function De(e) {
        switch (e) {
        case "svg":
            return "http://www.w3.org/2000/svg";
        case "math":
            return "http://www.w3.org/1998/Math/MathML";
        default:
            return "http://www.w3.org/1999/xhtml"
        }
    }
    function Le(e, t) {
        return null == e || "http://www.w3.org/1999/xhtml" === e ? De(t) : "http://www.w3.org/2000/svg" === e &&
"foreignObject" === t ? "http://www.w3.org/1999/xhtml" : e
    }
    var Ae, Ue, Ve = (Ue = function(e, t) {
        if (e.namespaceURI !== Fe || "innerHTML"in e)
            e.innerHTML = t;
        else {
            for ((Ae = Ae || document.createElement("div")).innerHTML = "<svg>" + t.valueOf().toString() +
"</svg>",
            t = Ae.firstChild; e.firstChild; )
                e.removeChild(e.firstChild);
            for (; t.firstChild; )
                e.appendChild(t.firstChild)
        }
    }
    ,
    "undefined" !== typeof MSApp && MSApp.execUnsafeLocalFunction ? function(e, t, n, r) {
        MSApp.execUnsafeLocalFunction((function() {
            return Ue(e, t)
        }
        ))
    }
    : Ue);
    function We(e, t) {
        if (t) {
            var n = e.firstChild;
            if (n && n === e.lastChild && 3 === n.nodeType)
                return void (n.nodeValue = t)
        }
        e.textContent = t
    }
    function je(e, t) {
        var n = {};
        return n[e.toLowerCase()] = t.toLowerCase(),
        n["Webkit" + e] = "webkit" + t,
```

```
            n["Moz" + e] = "moz" + t,
            n
        }
        var Qe = {
            animationend: je("Animation", "AnimationEnd"),
            animationiteration: je("Animation", "AnimationIteration"),
            animationstart: je("Animation", "AnimationStart"),
            transitionend: je("Transition", "TransitionEnd")
        }
          , $e = {}
          , Be = {};
        function He(e) {
            if ($e[e])
                return $e[e];
            if (!Qe[e])
                return e;
            var t, n = Qe[e];
            for (t in n)
                if (n.hasOwnProperty(t) && t in Be)
                    return $e[e] = n[t];
            return e
        }
        _ && (Be = document.createElement("div").style,
        "AnimationEvent"in window || (delete Qe.animationend.animation,
        delete Qe.animationiteration.animation,
        delete Qe.animationstart.animation),
        "TransitionEvent"in window || delete Qe.transitionend.transition);
        var Ke = He("animationend")
          , qe = He("animationiteration")
          , Ye = He("animationstart")
          , Xe = He("transitionend")
          , Ge = "abort canplay canplaythrough durationchange emptied encrypted ended error loadeddata
loadedmetadata loadstart pause play playing progress ratechange seeked seeking stalled suspend timeupdate
volumechange waiting".split(" ")
          , Ze = new ("function" === typeof WeakMap ? WeakMap : Map);
        function Je(e) {
            var t = Ze.get(e);
            return void 0 === t && (t = new Map,
            Ze.set(e, t)),
            t
        }
        function et(e) {
            var t = e
              , n = e;
            if (e.alternate)
                for (; t.return; )
                    t = t.return;
            else {
                e = t;
                do {
                    0 !== (1026 & (t = e).effectTag) && (n = t.return),
                    e = t.return
                } while (e)
            }
            return 3 === t.tag ? n : null
        }
        function tt(e) {
            if (13 === e.tag) {
                var t = e.memoizedState;
                if (null === t && (null !== (e = e.alternate) && (t = e.memoizedState)),
                null !== t)
                    return t.dehydrated
            }
            return null
        }
        function nt(e) {
            if (et(e) !== e)
                throw Error(a(188))
        }
```

```
function rt(e) {
    if (!(e = function(e) {
        var t = e.alternate;
        if (!t) {
            if (null === (t = et(e)))
                throw Error(a(188));
            return t !== e ? null : e
        }
        for (var n = e, r = t; ; ) {
            var l = n.return;
            if (null === l)
                break;
            var i = l.alternate;
            if (null === i) {
                if (null !== (r = l.return)) {
                    n = r;
                    continue
                }
                break
            }
            if (l.child === i.child) {
                for (i = l.child; i; ) {
                    if (i === n)
                        return nt(l),
                        e;
                    if (i === r)
                        return nt(l),
                        t;
                    i = i.sibling
                }
                throw Error(a(188))
            }
            if (n.return !== r.return)
                n = l,
                r = i;
            else {
                for (var o = !1, u = l.child; u; ) {
                    if (u === n) {
                        o = !0,
                        n = l,
                        r = i;
                        break
                    }
                    if (u === r) {
                        o = !0,
                        r = l,
                        n = i;
                        break
                    }
                    u = u.sibling
                }
                if (!o) {
                    for (u = i.child; u; ) {
                        if (u === n) {
                            o = !0,
                            n = i,
                            r = l;
                            break
                        }
                        if (u === r) {
                            o = !0,
                            r = i,
                            n = l;
                            break
                        }
                        u = u.sibling
                    }
                    if (!o)
                        throw Error(a(189))
```

```
                }
            }
            if (n.alternate !== r)
                throw Error(a(190))
        }
        if (3 !== n.tag)
            throw Error(a(188));
        return n.stateNode.current === n ? e : t
    }(e)))
        return null;
    for (var t = e; ; ) {
        if (5 === t.tag || 6 === t.tag)
            return t;
        if (t.child)
            t.child.return = t,
            t = t.child;
        else {
            if (t === e)
                break;
            for (; !t.sibling; ) {
                if (!t.return || t.return === e)
                    return null;
                t = t.return
            }
            t.sibling.return = t.return,
            t = t.sibling
        }
    }
    return null
}
function lt(e, t) {
    if (null == t)
        throw Error(a(30));
    return null == e ? t : Array.isArray(e) ? Array.isArray(t) ? (e.push.apply(e, t),
    e) : (e.push(t),
    e) : Array.isArray(t) ? [e].concat(t) : [e, t]
}
function it(e, t, n) {
    Array.isArray(e) ? e.forEach(t, n) : e && t.call(n, e)
}
var at = null;
function ot(e) {
    if (e) {
        var t = e._dispatchListeners
          , n = e._dispatchInstances;
        if (Array.isArray(t))
            for (var r = 0; r < t.length && !e.isPropagationStopped(); r++)
                y(e, t[r], n[r]);
        else
            t && y(e, t, n);
        e._dispatchListeners = null,
        e._dispatchInstances = null,
        e.isPersistent() || e.constructor.release(e)
    }
}
function ut(e) {
    if (null !== e && (at = lt(at, e)),
    e = at,
    at = null,
    e) {
        if (it(e, ot),
        at)
            throw Error(a(95));
        if (s)
            throw e = f,
            s = !1,
            f = null,
            e
    }
```

```
        }
        function ct(e) {
            return (e = e.target || e.srcElement || window).correspondingUseElement && (e =
e.correspondingUseElement),
            3 === e.nodeType ? e.parentNode : e
        }
        function st(e) {
            if (!_)
                return !1;
            var t = (e = "on" + e)in document;
            return t || ((t = document.createElement("div")).setAttribute(e, "return;"),
            t = "function" === typeof t[e]),
            t
        }
        var ft = [];
        function dt(e) {
            e.topLevelType = null,
            e.nativeEvent = null,
            e.targetInst = null,
            e.ancestors.length = 0,
            10 > ft.length && ft.push(e)
        }
        function pt(e, t, n, r) {
            if (ft.length) {
                var l = ft.pop();
                return l.topLevelType = e,
                l.eventSystemFlags = r,
                l.nativeEvent = t,
                l.targetInst = n,
                l
            }
            return {
                topLevelType: e,
                eventSystemFlags: r,
                nativeEvent: t,
                targetInst: n,
                ancestors: []
            }
        }
        function mt(e) {
            var t = e.targetInst
              , n = t;
            do {
                if (!n) {
                    e.ancestors.push(n);
                    break
                }
                var r = n;
                if (3 === r.tag)
                    r = r.stateNode.containerInfo;
                else {
                    for (; r.return; )
                        r = r.return;
                    r = 3 !== r.tag ? null : r.stateNode.containerInfo
                }
                if (!r)
                    break;
                5 !== (t = n.tag) && 6 !== t || e.ancestors.push(n),
                n = On(r)
            } while (n);
            for (n = 0; n < e.ancestors.length; n++) {
                t = e.ancestors[n];
                var l = ct(e.nativeEvent);
                r = e.topLevelType;
                var i = e.nativeEvent
                  , a = e.eventSystemFlags;
                0 === n && (a |= 64);
                for (var o = null, u = 0; u < x.length; u++) {
                    var c = x[u];
```

```
                c && (c = c.extractEvents(r, t, i, l, a)) && (o = lt(o, c))
            }
            ut(o)
        }
    }
    function ht(e, t, n) {
        if (!n.has(e)) {
            switch (e) {
            case "scroll":
                Yt(t, "scroll", !0);
                break;
            case "focus":
            case "blur":
                Yt(t, "focus", !0),
                Yt(t, "blur", !0),
                n.set("blur", null),
                n.set("focus", null);
                break;
            case "cancel":
            case "close":
                st(e) && Yt(t, e, !0);
                break;
            case "invalid":
            case "submit":
            case "reset":
                break;
            default:
                -1 === Ge.indexOf(e) && qt(e, t)
            }
            n.set(e, null)
        }
    }
    var vt, yt, gt, bt = !1, wt = [], kt = null, xt = null, Tt = null, Et = new Map, St = new Map, Ct = [], _t =
"mousedown mouseup touchcancel touchend touchstart auxclick dblclick pointercancel pointerdown pointerup dragend
dragstart drop compositionend compositionstart keydown keypress keyup input textInput close cancel copy cut paste
click change contextmenu reset submit".split(" "), Pt = "focus blur dragenter dragleave mouseover mouseout
pointerover pointerout gotpointercapture lostpointercapture".split(" ");
    function Nt(e, t, n, r, l) {
        return {
            blockedOn: e,
            topLevelType: t,
            eventSystemFlags: 32 | n,
            nativeEvent: l,
            container: r
        }
    }
    function zt(e, t) {
        switch (e) {
        case "focus":
        case "blur":
            kt = null;
            break;
        case "dragenter":
        case "dragleave":
            xt = null;
            break;
        case "mouseover":
        case "mouseout":
            Tt = null;
            break;
        case "pointerover":
        case "pointerout":
            Et.delete(t.pointerId);
            break;
        case "gotpointercapture":
        case "lostpointercapture":
            St.delete(t.pointerId)
        }
    }
```

```
function Ot(e, t, n, r, l, i) {
    return null === e || e.nativeEvent !== i ? (e = Nt(t, n, r, l, i),
    null !== t && (null !== (t = Rn(t)) && yt(t)),
    e) : (e.eventSystemFlags |= r,
    e)
}
function Rt(e) {
    var t = On(e.target);
    if (null !== t) {
        var n = et(t);
        if (null !== n)
            if (13 === (t = n.tag)) {
                if (null !== (t = tt(n)))
                    return e.blockedOn = t,
                    void i.unstable_runWithPriority(e.priority, (function() {
                        gt(n)
                    }
                    ))
            } else if (3 === t && n.stateNode.hydrate)
                return void (e.blockedOn = 3 === n.tag ? n.stateNode.containerInfo : null)
    }
    e.blockedOn = null
}
function It(e) {
    if (null !== e.blockedOn)
        return !1;
    var t = Jt(e.topLevelType, e.eventSystemFlags, e.container, e.nativeEvent);
    if (null !== t) {
        var n = Rn(t);
        return null !== n && yt(n),
        e.blockedOn = t,
        !1
    }
    return !0
}
function Mt(e, t, n) {
    It(e) && n.delete(t)
}
function Ft() {
    for (bt = !1; 0 < wt.length; ) {
        var e = wt[0];
        if (null !== e.blockedOn) {
            null !== (e = Rn(e.blockedOn)) && vt(e);
            break
        }
        var t = Jt(e.topLevelType, e.eventSystemFlags, e.container, e.nativeEvent);
        null !== t ? e.blockedOn = t : wt.shift()
    }
    null !== kt && It(kt) && (kt = null),
    null !== xt && It(xt) && (xt = null),
    null !== Tt && It(Tt) && (Tt = null),
    Et.forEach(Mt),
    St.forEach(Mt)
}
function Dt(e, t) {
    e.blockedOn === t && (e.blockedOn = null,
    bt || (bt = !0,
    i.unstable_scheduleCallback(i.unstable_NormalPriority, Ft)))
}
function Lt(e) {
    function t(t) {
        return Dt(t, e)
    }
    if (0 < wt.length) {
        Dt(wt[0], e);
        for (var n = 1; n < wt.length; n++) {
            var r = wt[n];
            r.blockedOn === e && (r.blockedOn = null)
        }
```

```
            }
            for (null !== kt && Dt(kt, e),
            null !== xt && Dt(xt, e),
            null !== Tt && Dt(Tt, e),
            Et.forEach(t),
            St.forEach(t),
            n = 0; n < Ct.length; n++)
                (r = Ct[n]).blockedOn === e && (r.blockedOn = null);
            for (; 0 < Ct.length && null === (n = Ct[0]).blockedOn; )
                Rt(n),
                null === n.blockedOn && Ct.shift()
        }
        var At = {}
          , Ut = new Map
          , Vt = new Map
          , Wt = ["abort", "abort", Ke, "animationEnd", qe, "animationIteration", Ye, "animationStart", "canplay",
"canPlay", "canplaythrough", "canPlayThrough", "durationchange", "durationChange", "emptied", "emptied",
"encrypted", "encrypted", "ended", "ended", "error", "error", "gotpointercapture", "gotPointerCapture", "load",
"load", "loadeddata", "loadedData", "loadedmetadata", "loadedMetadata", "loadstart", "loadStart",
"lostpointercapture", "lostPointerCapture", "playing", "playing", "progress", "progress", "seeking", "seeking",
"stalled", "stalled", "suspend", "suspend", "timeupdate", "timeUpdate", Xe, "transitionEnd", "waiting", "waiting"];
        function jt(e, t) {
            for (var n = 0; n < e.length; n += 2) {
                var r = e[n]
                  , l = e[n + 1]
                  , i = "on" + (l[0].toUpperCase() + l.slice(1));
                i = {
                    phasedRegistrationNames: {
                        bubbled: i,
                        captured: i + "Capture"
                    },
                    dependencies: [r],
                    eventPriority: t
                },
                Vt.set(r, t),
                Ut.set(r, i),
                At[l] = i
            }
        }
        jt("blur blur cancel cancel click click close close contextmenu contextMenu copy copy cut cut auxclick
auxClick dblclick doubleClick dragend dragEnd dragstart dragStart drop drop focus focus input input invalid invalid
keydown keyDown keypress keyPress keyup keyUp mousedown mouseDown mouseup mouseUp paste paste pause pause play play
pointercancel pointerCancel pointerdown pointerDown pointerup pointerUp ratechange rateChange reset reset seeked
seeked submit submit touchcancel touchCancel touchend touchEnd touchstart touchStart volumechange
volumeChange".split(" "), 0),
        jt("drag drag dragenter dragEnter dragexit dragExit dragleave dragLeave dragover dragOver mousemove
mouseMove mouseout mouseOut mouseover mouseOver pointermove pointerMove pointerout pointerOut pointerover
pointerOver scroll scroll toggle toggle touchmove touchMove wheel wheel".split(" "), 1),
        jt(Wt, 2);
        for (var Qt = "change selectionchange textInput compositionstart compositionend compositionupdate".split("
"), $t = 0; $t < Qt.length; $t++)
            Vt.set(Qt[$t], 0);
        var Bt = i.unstable_UserBlockingPriority
          , Ht = i.unstable_runWithPriority
          , Kt = !0;
        function qt(e, t) {
            Yt(t, e, !1)
        }
        function Yt(e, t, n) {
            var r = Vt.get(t);
            switch (void 0 === r ? 2 : r) {
            case 0:
                r = Xt.bind(null, t, 1, e);
                break;
            case 1:
                r = Gt.bind(null, t, 1, e);
                break;
            default:
                r = Zt.bind(null, t, 1, e)
```

```
    }
    n ? e.addEventListener(t, r, !0) : e.addEventListener(t, r, !1)
}
function Xt(e, t, n, r) {
    A || D();
    var l = Zt
      , i = A;
    A = !0;
    try {
        F(l, e, t, n, r)
    } finally {
        (A = i) || V()
    }
}
function Gt(e, t, n, r) {
    Ht(Bt, Zt.bind(null, e, t, n, r))
}
function Zt(e, t, n, r) {
    if (Kt)
        if (0 < wt.length && -1 < _t.indexOf(e))
            e = Nt(null, e, t, n, r),
            wt.push(e);
        else {
            var l = Jt(e, t, n, r);
            if (null === l)
                zt(e, r);
            else if (-1 < _t.indexOf(e))
                e = Nt(l, e, t, n, r),
                wt.push(e);
            else if (!function(e, t, n, r, l) {
                switch (t) {
                case "focus":
                    return kt = Ot(kt, e, t, n, r, l),
                    !0;
                case "dragenter":
                    return xt = Ot(xt, e, t, n, r, l),
                    !0;
                case "mouseover":
                    return Tt = Ot(Tt, e, t, n, r, l),
                    !0;
                case "pointerover":
                    var i = l.pointerId;
                    return Et.set(i, Ot(Et.get(i) || null, e, t, n, r, l)),
                    !0;
                case "gotpointercapture":
                    return i = l.pointerId,
                    St.set(i, Ot(St.get(i) || null, e, t, n, r, l)),
                    !0
                }
                return !1
            }(l, e, t, n, r)) {
                zt(e, r),
                e = pt(e, r, null, t);
                try {
                    W(mt, e)
                } finally {
                    dt(e)
                }
            }
        }
}
function Jt(e, t, n, r) {
    if (null !== (n = On(n = ct(r)))) {
        var l = et(n);
        if (null === l)
            n = null;
        else {
            var i = l.tag;
            if (13 === i) {
```

```
                    if (null !== (n = tt(l)))
                        return n;
                    n = null
                } else if (3 === i) {
                    if (l.stateNode.hydrate)
                        return 3 === l.tag ? l.stateNode.containerInfo : null;
                    n = null
                } else
                    l !== n && (n = null)
            }
        }
        e = pt(e, r, n, t);
        try {
            W(mt, e)
        } finally {
            dt(e)
        }
        return null
    }
    var en = {
        animationIterationCount: !0,
        borderImageOutset: !0,
        borderImageSlice: !0,
        borderImageWidth: !0,
        boxFlex: !0,
        boxFlexGroup: !0,
        boxOrdinalGroup: !0,
        columnCount: !0,
        columns: !0,
        flex: !0,
        flexGrow: !0,
        flexPositive: !0,
        flexShrink: !0,
        flexNegative: !0,
        flexOrder: !0,
        gridArea: !0,
        gridRow: !0,
        gridRowEnd: !0,
        gridRowSpan: !0,
        gridRowStart: !0,
        gridColumn: !0,
        gridColumnEnd: !0,
        gridColumnSpan: !0,
        gridColumnStart: !0,
        fontWeight: !0,
        lineClamp: !0,
        lineHeight: !0,
        opacity: !0,
        order: !0,
        orphans: !0,
        tabSize: !0,
        widows: !0,
        zIndex: !0,
        zoom: !0,
        fillOpacity: !0,
        floodOpacity: !0,
        stopOpacity: !0,
        strokeDasharray: !0,
        strokeDashoffset: !0,
        strokeMiterlimit: !0,
        strokeOpacity: !0,
        strokeWidth: !0
    }
      , tn = ["Webkit", "ms", "Moz", "O"];
    function nn(e, t, n) {
        return null == t || "boolean" === typeof t || "" === t ? "" : n || "number" !== typeof t || 0 === t ||
en.hasOwnProperty(e) && en[e] ? ("" + t).trim() : t + "px"
    }
    function rn(e, t) {
```

```
    for (var n in e = e.style,
    t)
        if (t.hasOwnProperty(n)) {
            var r = 0 === n.indexOf("--")
              , l = nn(n, t[n], r);
            "float" === n && (n = "cssFloat"),
            r ? e.setProperty(n, l) : e[n] = l
        }
}
Object.keys(en).forEach((function(e) {
    tn.forEach((function(t) {
        t = t + e.charAt(0).toUpperCase() + e.substring(1),
        en[t] = en[e]
    }
    ))
}
));
var ln = l({
    menuitem: !0
}, {
    area: !0,
    base: !0,
    br: !0,
    col: !0,
    embed: !0,
    hr: !0,
    img: !0,
    input: !0,
    keygen: !0,
    link: !0,
    meta: !0,
    param: !0,
    source: !0,
    track: !0,
    wbr: !0
});
function an(e, t) {
    if (t) {
        if (ln[e] && (null != t.children || null != t.dangerouslySetInnerHTML))
            throw Error(a(137, e, ""));
        if (null != t.dangerouslySetInnerHTML) {
            if (null != t.children)
                throw Error(a(60));
            if ("object" !== typeof t.dangerouslySetInnerHTML || !("__html"in t.dangerouslySetInnerHTML))
                throw Error(a(61))
        }
        if (null != t.style && "object" !== typeof t.style)
            throw Error(a(62, ""))
    }
}
function on(e, t) {
    if (-1 === e.indexOf("-"))
        return "string" === typeof t.is;
    switch (e) {
    case "annotation-xml":
    case "color-profile":
    case "font-face":
    case "font-face-src":
    case "font-face-uri":
    case "font-face-format":
    case "font-face-name":
    case "missing-glyph":
        return !1;
    default:
        return !0
    }
}
var un = Me;
function cn(e, t) {
```

```
            var n = Je(e = 9 === e.nodeType || 11 === e.nodeType ? e : e.ownerDocument);
            t = S[t];
            for (var r = 0; r < t.length; r++)
                ht(t[r], e, n)
        }
        function sn() {}
        function fn(e) {
            if ("undefined" === typeof (e = e || ("undefined" !== typeof document ? document : void 0)))
                return null;
            try {
                return e.activeElement || e.body
            } catch (t) {
                return e.body
            }
        }
        function dn(e) {
            for (; e && e.firstChild; )
                e = e.firstChild;
            return e
        }
        function pn(e, t) {
            var n, r = dn(e);
            for (e = 0; r; ) {
                if (3 === r.nodeType) {
                    if (n = e + r.textContent.length,
                    e <= t && n >= t)
                        return {
                            node: r,
                            offset: t - e
                        };
                    e = n
                }
                e: {
                    for (; r; ) {
                        if (r.nextSibling) {
                            r = r.nextSibling;
                            break e
                        }
                        r = r.parentNode
                    }
                    r = void 0
                }
                r = dn(r)
            }
        }
        function mn(e, t) {
            return !(!e || !t) && (e === t || (!e || 3 !== e.nodeType) && (t && 3 === t.nodeType ? mn(e,
t.parentNode) : "contains"in e ? e.contains(t) : !!e.compareDocumentPosition && !!(16 &
e.compareDocumentPosition(t))))
        }
        function hn() {
            for (var e = window, t = fn(); t instanceof e.HTMLIFrameElement; ) {
                try {
                    var n = "string" === typeof t.contentWindow.location.href
                } catch (r) {
                    n = !1
                }
                if (!n)
                    break;
                t = fn((e = t.contentWindow).document)
            }
            return t
        }
        function vn(e) {
            var t = e && e.nodeName && e.nodeName.toLowerCase();
            return t && ("input" === t && ("text" === e.type || "search" === e.type || "tel" === e.type || "url" ===
e.type || "password" === e.type) || "textarea" === t || "true" === e.contentEditable)
        }
        var yn = "$?"
```

```
          , gn = "$!"
          , bn = null
          , wn = null;
        function kn(e, t) {
            switch (e) {
            case "button":
            case "input":
            case "select":
            case "textarea":
                return !!t.autoFocus
            }
            return !1
        }
        function xn(e, t) {
            return "textarea" === e || "option" === e || "noscript" === e || "string" === typeof t.children ||
"number" === typeof t.children || "object" === typeof t.dangerouslySetInnerHTML && null !==
t.dangerouslySetInnerHTML && null != t.dangerouslySetInnerHTML.__html
        }
        var Tn = "function" === typeof setTimeout ? setTimeout : void 0
          , En = "function" === typeof clearTimeout ? clearTimeout : void 0;
        function Sn(e) {
            for (; null != e; e = e.nextSibling) {
                var t = e.nodeType;
                if (1 === t || 3 === t)
                    break
            }
            return e
        }
        function Cn(e) {
            e = e.previousSibling;
            for (var t = 0; e; ) {
                if (8 === e.nodeType) {
                    var n = e.data;
                    if ("$" === n || n === gn || n === yn) {
                        if (0 === t)
                            return e;
                        t--
                    } else
                        "/$" === n && t++
                }
                e = e.previousSibling
            }
            return null
        }
        var _n = Math.random().toString(36).slice(2)
          , Pn = "__reactInternalInstance$" + _n
          , Nn = "__reactEventHandlers$" + _n
          , zn = "__reactContainere$" + _n;
        function On(e) {
            var t = e[Pn];
            if (t)
                return t;
            for (var n = e.parentNode; n; ) {
                if (t = n[zn] || n[Pn]) {
                    if (n = t.alternate,
                    null !== t.child || null !== n && null !== n.child)
                        for (e = Cn(e); null !== e; ) {
                            if (n = e[Pn])
                                return n;
                            e = Cn(e)
                        }
                    return t
                }
                n = (e = n).parentNode
            }
            return null
        }
        function Rn(e) {
            return !(e = e[Pn] || e[zn]) || 5 !== e.tag && 6 !== e.tag && 13 !== e.tag && 3 !== e.tag ? null : e
```

```
        }
        function In(e) {
            if (5 === e.tag || 6 === e.tag)
                return e.stateNode;
            throw Error(a(33))
        }
        function Mn(e) {
            return e[Nn] || null
        }
        function Fn(e) {
            do {
                e = e.return
            } while (e && 5 !== e.tag);
            return e || null
        }
        function Dn(e, t) {
            var n = e.stateNode;
            if (!n)
                return null;
            var r = m(n);
            if (!r)
                return null;
            n = r[t];
            e: switch (t) {
            case "onClick":
            case "onClickCapture":
            case "onDoubleClick":
            case "onDoubleClickCapture":
            case "onMouseDown":
            case "onMouseDownCapture":
            case "onMouseMove":
            case "onMouseMoveCapture":
            case "onMouseUp":
            case "onMouseUpCapture":
            case "onMouseEnter":
                (r = !r.disabled) || (r = !("button" === (e = e.type) || "input" === e || "select" === e ||
"textarea" === e)),
                e = !r;
                break e;
            default:
                e = !1
            }
            if (e)
                return null;
            if (n && "function" !== typeof n)
                throw Error(a(231, t, typeof n));
            return n
        }
        function Ln(e, t, n) {
            (t = Dn(e, n.dispatchConfig.phasedRegistrationNames[t])) && (n._dispatchListeners =
lt(n._dispatchListeners, t),
            n._dispatchInstances = lt(n._dispatchInstances, e))
        }
        function An(e) {
            if (e && e.dispatchConfig.phasedRegistrationNames) {
                for (var t = e._targetInst, n = []; t; )
                    n.push(t),
                    t = Fn(t);
                for (t = n.length; 0 < t--; )
                    Ln(n[t], "captured", e);
                for (t = 0; t < n.length; t++)
                    Ln(n[t], "bubbled", e)
            }
        }
        function Un(e, t, n) {
            e && n && n.dispatchConfig.registrationName && (t = Dn(e, n.dispatchConfig.registrationName)) &&
(n._dispatchListeners = lt(n._dispatchListeners, t),
            n._dispatchInstances = lt(n._dispatchInstances, e))
        }
```

```
        function Vn(e) {
            e && e.dispatchConfig.registrationName && Un(e._targetInst, null, e)
        }
        function Wn(e) {
            it(e, An)
        }
        var jn = null
          , Qn = null
          , $n = null;
        function Bn() {
            if ($n)
                return $n;
            var e, t, n = Qn, r = n.length, l = "value"in jn ? jn.value : jn.textContent, i = l.length;
            for (e = 0; e < r && n[e] === l[e]; e++)
                ;
            var a = r - e;
            for (t = 1; t <= a && n[r - t] === l[i - t]; t++)
                ;
            return $n = l.slice(e, 1 < t ? 1 - t : void 0)
        }
        function Hn() {
            return !0
        }
        function Kn() {
            return !1
        }
        function qn(e, t, n, r) {
            for (var l in this.dispatchConfig = e,
            this._targetInst = t,
            this.nativeEvent = n,
            e = this.constructor.Interface)
                e.hasOwnProperty(l) && ((t = e[l]) ? this[l] = t(n) : "target" === l ? this.target = r : this[l] =
n[l]);
            return this.isDefaultPrevented = (null != n.defaultPrevented ? n.defaultPrevented : !1 ===
n.returnValue) ? Hn : Kn,
            this.isPropagationStopped = Kn,
            this
        }
        function Yn(e, t, n, r) {
            if (this.eventPool.length) {
                var l = this.eventPool.pop();
                return this.call(l, e, t, n, r),
                l
            }
            return new this(e,t,n,r)
        }
        function Xn(e) {
            if (!(e instanceof this))
                throw Error(a(279));
            e.destructor(),
            10 > this.eventPool.length && this.eventPool.push(e)
        }
        function Gn(e) {
            e.eventPool = [],
            e.getPooled = Yn,
            e.release = Xn
        }
        l(qn.prototype, {
            preventDefault: function() {
                this.defaultPrevented = !0;
                var e = this.nativeEvent;
                e && (e.preventDefault ? e.preventDefault() : "unknown" !== typeof e.returnValue && (e.returnValue =
!1),
                this.isDefaultPrevented = Hn)
            },
            stopPropagation: function() {
                var e = this.nativeEvent;
                e && (e.stopPropagation ? e.stopPropagation() : "unknown" !== typeof e.cancelBubble &&
(e.cancelBubble = !0),
```

```
        this.isPropagationStopped = Hn)
    },
    persist: function() {
        this.isPersistent = Hn
    },
    isPersistent: Kn,
    destructor: function() {
        var e, t = this.constructor.Interface;
        for (e in t)
            this[e] = null;
        this.nativeEvent = this._targetInst = this.dispatchConfig = null,
        this.isPropagationStopped = this.isDefaultPrevented = Kn,
        this._dispatchInstances = this._dispatchListeners = null
    }
}),
qn.Interface = {
    type: null,
    target: null,
    currentTarget: function() {
        return null
    },
    eventPhase: null,
    bubbles: null,
    cancelable: null,
    timeStamp: function(e) {
        return e.timeStamp || Date.now()
    },
    defaultPrevented: null,
    isTrusted: null
},
qn.extend = function(e) {
    function t() {}
    function n() {
        return r.apply(this, arguments)
    }
    var r = this;
    t.prototype = r.prototype;
    var i = new t;
    return l(i, n.prototype),
    n.prototype = i,
    n.prototype.constructor = n,
    n.Interface = l({}, r.Interface, e),
    n.extend = r.extend,
    Gn(n),
    n
}
,
Gn(qn);
var Zn = qn.extend({
    data: null
})
  , Jn = qn.extend({
    data: null
})
  , er = [9, 13, 27, 32]
  , tr = _ && "CompositionEvent"in window
  , nr = null;
_ && "documentMode"in document && (nr = document.documentMode);
var rr = _ && "TextEvent"in window && !nr
  , lr = _ && (!tr || nr && 8 < nr && 11 >= nr)
  , ir = String.fromCharCode(32)
  , ar = {
    beforeInput: {
        phasedRegistrationNames: {
            bubbled: "onBeforeInput",
            captured: "onBeforeInputCapture"
        },
        dependencies: ["compositionend", "keypress", "textInput", "paste"]
    },
```

```
            compositionEnd: {
                phasedRegistrationNames: {
                    bubbled: "onCompositionEnd",
                    captured: "onCompositionEndCapture"
                },
                dependencies: "blur compositionend keydown keypress keyup mousedown".split(" ")
            },
            compositionStart: {
                phasedRegistrationNames: {
                    bubbled: "onCompositionStart",
                    captured: "onCompositionStartCapture"
                },
                dependencies: "blur compositionstart keydown keypress keyup mousedown".split(" ")
            },
            compositionUpdate: {
                phasedRegistrationNames: {
                    bubbled: "onCompositionUpdate",
                    captured: "onCompositionUpdateCapture"
                },
                dependencies: "blur compositionupdate keydown keypress keyup mousedown".split(" ")
            }
        }
          , or = !1;
        function ur(e, t) {
            switch (e) {
            case "keyup":
                return -1 !== er.indexOf(t.keyCode);
            case "keydown":
                return 229 !== t.keyCode;
            case "keypress":
            case "mousedown":
            case "blur":
                return !0;
            default:
                return !1
            }
        }
        function cr(e) {
            return "object" === typeof (e = e.detail) && "data"in e ? e.data : null
        }
        var sr = !1;
        var fr = {
            eventTypes: ar,
            extractEvents: function(e, t, n, r) {
                var l;
                if (tr)
                    e: {
                        switch (e) {
                        case "compositionstart":
                            var i = ar.compositionStart;
                            break e;
                        case "compositionend":
                            i = ar.compositionEnd;
                            break e;
                        case "compositionupdate":
                            i = ar.compositionUpdate;
                            break e
                        }
                        i = void 0
                    }
                else
                    sr ? ur(e, n) && (i = ar.compositionEnd) : "keydown" === e && 229 === n.keyCode && (i =
ar.compositionStart);
                return i ? (lr && "ko" !== n.locale && (sr || i !== ar.compositionStart ? i === ar.compositionEnd &&
sr && (l = Bn()) : (Qn = "value"in (jn = r) ? jn.value : jn.textContent,
                sr = !0)),
                i = Zn.getPooled(i, t, n, r),
                l ? i.data = l : null !== (l = cr(n)) && (i.data = l),
                Wn(i),
```

```
        l = i) : l = null,
        (e = rr ? function(e, t) {
            switch (e) {
            case "compositionend":
                return cr(t);
            case "keypress":
                return 32 !== t.which ? null : (or = !0,
                ir);
            case "textInput":
                return (e = t.data) === ir && or ? null : e;
            default:
                return null
            }
        }(e, n) : function(e, t) {
            if (sr)
                return "compositionend" === e || !tr && ur(e, t) ? (e = Bn(),
                $n = Qn = jn = null,
                sr = !1,
                e) : null;
            switch (e) {
            case "paste":
                return null;
            case "keypress":
                if (!(t.ctrlKey || t.altKey || t.metaKey) || t.ctrlKey && t.altKey) {
                    if (t.char && 1 < t.char.length)
                        return t.char;
                    if (t.which)
                        return String.fromCharCode(t.which)
                }
                return null;
            case "compositionend":
                return lr && "ko" !== t.locale ? null : t.data;
            default:
                return null
            }
        }(e, n)) ? ((t = Jn.getPooled(ar.beforeInput, t, n, r)).data = e,
        Wn(t)) : t = null,
        null === l ? t : null === t ? l : [l, t]
    }
}
  , dr = {
    color: !0,
    date: !0,
    datetime: !0,
    "datetime-local": !0,
    email: !0,
    month: !0,
    number: !0,
    password: !0,
    range: !0,
    search: !0,
    tel: !0,
    text: !0,
    time: !0,
    url: !0,
    week: !0
};
function pr(e) {
    var t = e && e.nodeName && e.nodeName.toLowerCase();
    return "input" === t ? !!dr[e.type] : "textarea" === t
}
var mr = {
    change: {
        phasedRegistrationNames: {
            bubbled: "onChange",
            captured: "onChangeCapture"
        },
        dependencies: "blur change click focus input keydown keyup selectionchange".split(" ")
    }
```

```
};
function hr(e, t, n) {
    return (e = qn.getPooled(mr.change, e, t, n)).type = "change",
    R(n),
    Wn(e),
    e
}
var vr = null
  , yr = null;
function gr(e) {
    ut(e)
}
function br(e) {
    if (ke(In(e)))
        return e
}
function wr(e, t) {
    if ("change" === e)
        return t
}
var kr = !1;
function xr() {
    vr && (vr.detachEvent("onpropertychange", Tr),
    yr = vr = null)
}
function Tr(e) {
    if ("value" === e.propertyName && br(yr))
        if (e = hr(yr, e, ct(e)),
        A)
            ut(e);
        else {
            A = !0;
            try {
                M(gr, e)
            } finally {
                A = !1,
                V()
            }
        }
}
function Er(e, t, n) {
    "focus" === e ? (xr(),
    yr = n,
    (vr = t).attachEvent("onpropertychange", Tr)) : "blur" === e && xr()
}
function Sr(e) {
    if ("selectionchange" === e || "keyup" === e || "keydown" === e)
        return br(yr)
}
function Cr(e, t) {
    if ("click" === e)
        return br(t)
}
function _r(e, t) {
    if ("input" === e || "change" === e)
        return br(t)
}
_ && (kr = st("input") && (!document.documentMode || 9 < document.documentMode));
var Pr = {
    eventTypes: mr,
    _isInputEventSupported: kr,
    extractEvents: function(e, t, n, r) {
        var l = t ? In(t) : window
          , i = l.nodeName && l.nodeName.toLowerCase();
        if ("select" === i || "input" === i && "file" === l.type)
            var a = wr;
        else if (pr(l))
            if (kr)
                a = _r;
```

```
                    else {
                        a = Sr;
                        var o = Er
                    }
                else
                    (i = l.nodeName) && "input" === i.toLowerCase() && ("checkbox" === l.type || "radio" === l.type)
&& (a = Cr);
                if (a && (a = a(e, t)))
                    return hr(a, n, r);
                o && o(e, l, t),
                "blur" === e && (e = l._wrapperState) && e.controlled && "number" === l.type && _e(l, "number",
l.value)
            }
        }
          , Nr = qn.extend({
            view: null,
            detail: null
        })
          , zr = {
            Alt: "altKey",
            Control: "ctrlKey",
            Meta: "metaKey",
            Shift: "shiftKey"
        };
        function Or(e) {
            var t = this.nativeEvent;
            return t.getModifierState ? t.getModifierState(e) : !!(e = zr[e]) && !!t[e]
        }
        function Rr() {
            return Or
        }
        var Ir = 0
          , Mr = 0
          , Fr = !1
          , Dr = !1
          , Lr = Nr.extend({
            screenX: null,
            screenY: null,
            clientX: null,
            clientY: null,
            pageX: null,
            pageY: null,
            ctrlKey: null,
            shiftKey: null,
            altKey: null,
            metaKey: null,
            getModifierState: Rr,
            button: null,
            buttons: null,
            relatedTarget: function(e) {
                return e.relatedTarget || (e.fromElement === e.srcElement ? e.toElement : e.fromElement)
            },
            movementX: function(e) {
                if ("movementX"in e)
                    return e.movementX;
                var t = Ir;
                return Ir = e.screenX,
                Fr ? "mousemove" === e.type ? e.screenX - t : 0 : (Fr = !0,
                0)
            },
            movementY: function(e) {
                if ("movementY"in e)
                    return e.movementY;
                var t = Mr;
                return Mr = e.screenY,
                Dr ? "mousemove" === e.type ? e.screenY - t : 0 : (Dr = !0,
                0)
            }
        })
```

```
          , Ar = Lr.extend({
            pointerId: null,
            width: null,
            height: null,
            pressure: null,
            tangentialPressure: null,
            tiltX: null,
            tiltY: null,
            twist: null,
            pointerType: null,
            isPrimary: null
        })
          , Ur = {
            mouseEnter: {
                registrationName: "onMouseEnter",
                dependencies: ["mouseout", "mouseover"]
            },
            mouseLeave: {
                registrationName: "onMouseLeave",
                dependencies: ["mouseout", "mouseover"]
            },
            pointerEnter: {
                registrationName: "onPointerEnter",
                dependencies: ["pointerout", "pointerover"]
            },
            pointerLeave: {
                registrationName: "onPointerLeave",
                dependencies: ["pointerout", "pointerover"]
            }
        }
          , Vr = {
            eventTypes: Ur,
            extractEvents: function(e, t, n, r, l) {
                var i = "mouseover" === e || "pointerover" === e
                  , a = "mouseout" === e || "pointerout" === e;
                if (i && 0 === (32 & l) && (n.relatedTarget || n.fromElement) || !a && !i)
                    return null;
                (i = r.window === r ? r : (i = r.ownerDocument) ? i.defaultView || i.parentWindow : window,
                a) ? (a = t,
                null !== (t = (t = n.relatedTarget || n.toElement) ? On(t) : null) && (t !== et(t) || 5 !== t.tag &&
6 !== t.tag) && (t = null)) : a = null;
                if (a === t)
                    return null;
                if ("mouseout" === e || "mouseover" === e)
                    var o = Lr
                      , u = Ur.mouseLeave
                      , c = Ur.mouseEnter
                      , s = "mouse";
                else
                    "pointerout" !== e && "pointerover" !== e || (o = Ar,
                    u = Ur.pointerLeave,
                    c = Ur.pointerEnter,
                    s = "pointer");
                if (e = null == a ? i : In(a),
                i = null == t ? i : In(t),
                (u = o.getPooled(u, a, n, r)).type = s + "leave",
                u.target = e,
                u.relatedTarget = i,
                (n = o.getPooled(c, t, n, r)).type = s + "enter",
                n.target = i,
                n.relatedTarget = e,
                s = t,
                (r = a) && s)
                    e: {
                        for (c = s,
                        a = 0,
                        e = o = r; e; e = Fn(e))
                            a++;
                        for (e = 0,
```

```
                    t = c; t; t = Fn(t))
                        e++;
                    for (; 0 < a - e; )
                        o = Fn(o),
                        a--;
                    for (; 0 < e - a; )
                        c = Fn(c),
                        e--;
                    for (; a--; ) {
                        if (o === c || o === c.alternate)
                            break e;
                        o = Fn(o),
                        c = Fn(c)
                    }
                    o = null
                }
            else
                o = null;
            for (c = o,
            o = []; r && r !== c && (null === (a = r.alternate) || a !== c); )
                o.push(r),
                r = Fn(r);
            for (r = []; s && s !== c && (null === (a = s.alternate) || a !== c); )
                r.push(s),
                s = Fn(s);
            for (s = 0; s < o.length; s++)
                Un(o[s], "bubbled", u);
            for (s = r.length; 0 < s--; )
                Un(r[s], "captured", n);
            return 0 === (64 & l) ? [u] : [u, n]
        }
    };
    var Wr = "function" === typeof Object.is ? Object.is : function(e, t) {
        return e === t && (0 !== e || 1 / e === 1 / t) || e !== e && t !== t
    }
      , jr = Object.prototype.hasOwnProperty;
    function Qr(e, t) {
        if (Wr(e, t))
            return !0;
        if ("object" !== typeof e || null === e || "object" !== typeof t || null === t)
            return !1;
        var n = Object.keys(e)
          , r = Object.keys(t);
        if (n.length !== r.length)
            return !1;
        for (r = 0; r < n.length; r++)
            if (!jr.call(t, n[r]) || !Wr(e[n[r]], t[n[r]]))
                return !1;
        return !0
    }
    var $r = _ && "documentMode"in document && 11 >= document.documentMode
      , Br = {
        select: {
            phasedRegistrationNames: {
                bubbled: "onSelect",
                captured: "onSelectCapture"
            },
            dependencies: "blur contextmenu dragend focus keydown keyup mousedown mouseup
selectionchange".split(" ")
        }
    }
      , Hr = null
      , Kr = null
      , qr = null
      , Yr = !1;
    function Xr(e, t) {
        var n = t.window === t ? t.document : 9 === t.nodeType ? t : t.ownerDocument;
        return Yr || null == Hr || Hr !== fn(n) ? null : ("selectionStart"in (n = Hr) && vn(n) ? n = {
            start: n.selectionStart,
```

```
                end: n.selectionEnd
            } : n = {
                anchorNode: (n = (n.ownerDocument && n.ownerDocument.defaultView ||
window).getSelection()).anchorNode,
                anchorOffset: n.anchorOffset,
                focusNode: n.focusNode,
                focusOffset: n.focusOffset
            },
            qr && Qr(qr, n) ? null : (qr = n,
            (e = qn.getPooled(Br.select, Kr, e, t)).type = "select",
            e.target = Hr,
            Wn(e),
            e))
        }
        var Gr = {
            eventTypes: Br,
            extractEvents: function(e, t, n, r, l, i) {
                if (!(i = !(l = i || (r.window === r ? r.document : 9 === r.nodeType ? r : r.ownerDocument)))) {
                    e: {
                        l = Je(l),
                        i = S.onSelect;
                        for (var a = 0; a < i.length; a++)
                            if (!l.has(i[a])) {
                                l = !1;
                                break e
                            }
                        l = !0
                    }
                    i = !l
                }
                if (i)
                    return null;
                switch (l = t ? In(t) : window,
                e) {
                case "focus":
                    (pr(l) || "true" === l.contentEditable) && (Hr = l,
                    Kr = t,
                    qr = null);
                    break;
                case "blur":
                    qr = Kr = Hr = null;
                    break;
                case "mousedown":
                    Yr = !0;
                    break;
                case "contextmenu":
                case "mouseup":
                case "dragend":
                    return Yr = !1,
                    Xr(n, r);
                case "selectionchange":
                    if ($r)
                        break;
                case "keydown":
                case "keyup":
                    return Xr(n, r)
                }
                return null
            }
        }
          , Zr = qn.extend({
            animationName: null,
            elapsedTime: null,
            pseudoElement: null
        })
          , Jr = qn.extend({
            clipboardData: function(e) {
                return "clipboardData"in e ? e.clipboardData : window.clipboardData
            }
```

```
})
  , el = Nr.extend({
    relatedTarget: null
});
function tl(e) {
    var t = e.keyCode;
    return "charCode"in e ? 0 === (e = e.charCode) && 13 === t && (e = 13) : e = t,
    10 === e && (e = 13),
    32 <= e || 13 === e ? e : 0
}
var nl = {
    Esc: "Escape",
    Spacebar: " ",
    Left: "ArrowLeft",
    Up: "ArrowUp",
    Right: "ArrowRight",
    Down: "ArrowDown",
    Del: "Delete",
    Win: "OS",
    Menu: "ContextMenu",
    Apps: "ContextMenu",
    Scroll: "ScrollLock",
    MozPrintableKey: "Unidentified"
}
  , rl = {
    8: "Backspace",
    9: "Tab",
    12: "Clear",
    13: "Enter",
    16: "Shift",
    17: "Control",
    18: "Alt",
    19: "Pause",
    20: "CapsLock",
    27: "Escape",
    32: " ",
    33: "PageUp",
    34: "PageDown",
    35: "End",
    36: "Home",
    37: "ArrowLeft",
    38: "ArrowUp",
    39: "ArrowRight",
    40: "ArrowDown",
    45: "Insert",
    46: "Delete",
    112: "F1",
    113: "F2",
    114: "F3",
    115: "F4",
    116: "F5",
    117: "F6",
    118: "F7",
    119: "F8",
    120: "F9",
    121: "F10",
    122: "F11",
    123: "F12",
    144: "NumLock",
    145: "ScrollLock",
    224: "Meta"
}
  , ll = Nr.extend({
    key: function(e) {
        if (e.key) {
            var t = nl[e.key] || e.key;
            if ("Unidentified" !== t)
                return t
        }
```

```
            return "keypress" === e.type ? 13 === (e = tl(e)) ? "Enter" : String.fromCharCode(e) : "keydown" ===
e.type || "keyup" === e.type ? rl[e.keyCode] || "Unidentified" : ""
            },
            location: null,
            ctrlKey: null,
            shiftKey: null,
            altKey: null,
            metaKey: null,
            repeat: null,
            locale: null,
            getModifierState: Rr,
            charCode: function(e) {
                return "keypress" === e.type ? tl(e) : 0
            },
            keyCode: function(e) {
                return "keydown" === e.type || "keyup" === e.type ? e.keyCode : 0
            },
            which: function(e) {
                return "keypress" === e.type ? tl(e) : "keydown" === e.type || "keyup" === e.type ? e.keyCode : 0
            }
        })
          , il = Lr.extend({
            dataTransfer: null
        })
          , al = Nr.extend({
            touches: null,
            targetTouches: null,
            changedTouches: null,
            altKey: null,
            metaKey: null,
            ctrlKey: null,
            shiftKey: null,
            getModifierState: Rr
        })
          , ol = qn.extend({
            propertyName: null,
            elapsedTime: null,
            pseudoElement: null
        })
          , ul = Lr.extend({
            deltaX: function(e) {
                return "deltaX"in e ? e.deltaX : "wheelDeltaX"in e ? -e.wheelDeltaX : 0
            },
            deltaY: function(e) {
                return "deltaY"in e ? e.deltaY : "wheelDeltaY"in e ? -e.wheelDeltaY : "wheelDelta"in e ?
-e.wheelDelta : 0
            },
            deltaZ: null,
            deltaMode: null
        })
          , cl = {
            eventTypes: At,
            extractEvents: function(e, t, n, r) {
                var l = Ut.get(e);
                if (!l)
                    return null;
                switch (e) {
                case "keypress":
                    if (0 === tl(n))
                        return null;
                case "keydown":
                case "keyup":
                    e = ll;
                    break;
                case "blur":
                case "focus":
                    e = el;
                    break;
                case "click":
```

```
                    if (2 === n.button)
                        return null;
                case "auxclick":
                case "dblclick":
                case "mousedown":
                case "mousemove":
                case "mouseup":
                case "mouseout":
                case "mouseover":
                case "contextmenu":
                    e = Lr;
                    break;
                case "drag":
                case "dragend":
                case "dragenter":
                case "dragexit":
                case "dragleave":
                case "dragover":
                case "dragstart":
                case "drop":
                    e = il;
                    break;
                case "touchcancel":
                case "touchend":
                case "touchmove":
                case "touchstart":
                    e = al;
                    break;
                case Ke:
                case qe:
                case Ye:
                    e = Zr;
                    break;
                case Xe:
                    e = ol;
                    break;
                case "scroll":
                    e = Nr;
                    break;
                case "wheel":
                    e = ul;
                    break;
                case "copy":
                case "cut":
                case "paste":
                    e = Jr;
                    break;
                case "gotpointercapture":
                case "lostpointercapture":
                case "pointercancel":
                case "pointerdown":
                case "pointermove":
                case "pointerout":
                case "pointerover":
                case "pointerup":
                    e = Ar;
                    break;
                default:
                    e = qn
                }
                return Wn(t = e.getPooled(l, t, n, r)),
                t
            }
        };
        if (g)
            throw Error(a(101));
        g = Array.prototype.slice.call("ResponderEventPlugin SimpleEventPlugin EnterLeaveEventPlugin
ChangeEventPlugin SelectEventPlugin BeforeInputEventPlugin".split(" ")),
        w(),
```

```
m = Mn,
h = Rn,
v = In,
C({
    SimpleEventPlugin: cl,
    EnterLeaveEventPlugin: Vr,
    ChangeEventPlugin: Pr,
    SelectEventPlugin: Gr,
    BeforeInputEventPlugin: fr
});
var sl = []
  , fl = -1;
function dl(e) {
    0 > fl || (e.current = sl[fl],
    sl[fl] = null,
    fl--)
}
function pl(e, t) {
    fl++,
    sl[fl] = e.current,
    e.current = t
}
var ml = {}
  , hl = {
    current: ml
}
  , vl = {
    current: !1
}
  , yl = ml;
function gl(e, t) {
    var n = e.type.contextTypes;
    if (!n)
        return ml;
    var r = e.stateNode;
    if (r && r.__reactInternalMemoizedUnmaskedChildContext === t)
        return r.__reactInternalMemoizedMaskedChildContext;
    var l, i = {};
    for (l in n)
        i[l] = t[l];
    return r && ((e = e.stateNode).__reactInternalMemoizedUnmaskedChildContext = t,
    e.__reactInternalMemoizedMaskedChildContext = i),
    i
}
function bl(e) {
    return null !== (e = e.childContextTypes) && void 0 !== e
}
function wl() {
    dl(vl),
    dl(hl)
}
function kl(e, t, n) {
    if (hl.current !== ml)
        throw Error(a(168));
    pl(hl, t),
    pl(vl, n)
}
function xl(e, t, n) {
    var r = e.stateNode;
    if (e = t.childContextTypes,
    "function" !== typeof r.getChildContext)
        return n;
    for (var i in r = r.getChildContext())
        if (!(i in e))
            throw Error(a(108, ve(t) || "Unknown", i));
    return l({}, n, {}, r)
}
function Tl(e) {
    return e = (e = e.stateNode) && e.__reactInternalMemoizedMergedChildContext || ml,
```

```
    yl = hl.current,
    pl(hl, e),
    pl(vl, vl.current),
    !0
}
function El(e, t, n) {
    var r = e.stateNode;
    if (!r)
        throw Error(a(169));
    n ? (e = xl(e, t, yl),
    r.__reactInternalMemoizedMergedChildContext = e,
    dl(vl),
    dl(hl),
    pl(hl, e)) : dl(vl),
    pl(vl, n)
}
var Sl = i.unstable_runWithPriority
  , Cl = i.unstable_scheduleCallback
  , _l = i.unstable_cancelCallback
  , Pl = i.unstable_requestPaint
  , Nl = i.unstable_now
  , zl = i.unstable_getCurrentPriorityLevel
  , Ol = i.unstable_ImmediatePriority
  , Rl = i.unstable_UserBlockingPriority
  , Il = i.unstable_NormalPriority
  , Ml = i.unstable_LowPriority
  , Fl = i.unstable_IdlePriority
  , Dl = {}
  , Ll = i.unstable_shouldYield
  , Al = void 0 !== Pl ? Pl : function() {}
  , Ul = null
  , Vl = null
  , Wl = !1
  , jl = Nl()
  , Ql = 1e4 > jl ? Nl : function() {
    return Nl() - jl
}
;
function $l() {
    switch (zl()) {
    case Ol:
        return 99;
    case Rl:
        return 98;
    case Il:
        return 97;
    case Ml:
        return 96;
    case Fl:
        return 95;
    default:
        throw Error(a(332))
    }
}
function Bl(e) {
    switch (e) {
    case 99:
        return Ol;
    case 98:
        return Rl;
    case 97:
        return Il;
    case 96:
        return Ml;
    case 95:
        return Fl;
    default:
        throw Error(a(332))
    }
```

```
}
function Hl(e, t) {
    return e = Bl(e),
    Sl(e, t)
}
function Kl(e, t, n) {
    return e = Bl(e),
    Cl(e, t, n)
}
function ql(e) {
    return null === Ul ? (Ul = [e],
    Vl = Cl(Ol, Xl)) : Ul.push(e),
    Dl
}
function Yl() {
    if (null !== Vl) {
        var e = Vl;
        Vl = null,
        _l(e)
    }
    Xl()
}
function Xl() {
    if (!Wl && null !== Ul) {
        Wl = !0;
        var e = 0;
        try {
            var t = Ul;
            Hl(99, (function() {
                for (; e < t.length; e++) {
                    var n = t[e];
                    do {
                        n = n(!0)
                    } while (null !== n)
                }
            }
            )),
            Ul = null
        } catch (n) {
            throw null !== Ul && (Ul = Ul.slice(e + 1)),
            Cl(Ol, Yl),
            n
        } finally {
            Wl = !1
        }
    }
}
function Gl(e, t, n) {
    return 1073741821 - (1 + ((1073741821 - e + t / 10) / (n /= 10) | 0)) * n
}
function Zl(e, t) {
    if (e && e.defaultProps)
        for (var n in t = l({}, t),
        e = e.defaultProps)
            void 0 === t[n] && (t[n] = e[n]);
    return t
}
var Jl = {
    current: null
}
  , ei = null
  , ti = null
  , ni = null;
function ri() {
    ni = ti = ei = null
}
function li(e) {
    var t = Jl.current;
    dl(Jl),
```

```
    e.type._context._currentValue = t
}
function ii(e, t) {
    for (; null !== e; ) {
        var n = e.alternate;
        if (e.childExpirationTime < t)
            e.childExpirationTime = t,
            null !== n && n.childExpirationTime < t && (n.childExpirationTime = t);
        else {
            if (!(null !== n && n.childExpirationTime < t))
                break;
            n.childExpirationTime = t
        }
        e = e.return
    }
}
function ai(e, t) {
    ei = e,
    ni = ti = null,
    null !== (e = e.dependencies) && null !== e.firstContext && (e.expirationTime >= t && (Ia = !0),
    e.firstContext = null)
}
function oi(e, t) {
    if (ni !== e && !1 !== t && 0 !== t)
        if ("number" === typeof t && 1073741823 !== t || (ni = e,
        t = 1073741823),
        t = {
            context: e,
            observedBits: t,
            next: null
        },
        null === ti) {
            if (null === ei)
                throw Error(a(308));
            ti = t,
            ei.dependencies = {
                expirationTime: 0,
                firstContext: t,
                responders: null
            }
        } else
            ti = ti.next = t;
    return e._currentValue
}
var ui = !1;
function ci(e) {
    e.updateQueue = {
        baseState: e.memoizedState,
        baseQueue: null,
        shared: {
            pending: null
        },
        effects: null
    }
}
function si(e, t) {
    e = e.updateQueue,
    t.updateQueue === e && (t.updateQueue = {
        baseState: e.baseState,
        baseQueue: e.baseQueue,
        shared: e.shared,
        effects: e.effects
    })
}
function fi(e, t) {
    return (e = {
        expirationTime: e,
        suspenseConfig: t,
        tag: 0,
```

```
        payload: null,
        callback: null,
        next: null
    }).next = e
}
function di(e, t) {
    if (null !== (e = e.updateQueue)) {
        var n = (e = e.shared).pending;
        null === n ? t.next = t : (t.next = n.next,
        n.next = t),
        e.pending = t
    }
}
function pi(e, t) {
    var n = e.alternate;
    null !== n && si(n, e),
    null === (n = (e = e.updateQueue).baseQueue) ? (e.baseQueue = t.next = t,
    t.next = t) : (t.next = n.next,
    n.next = t)
}
function mi(e, t, n, r) {
    var i = e.updateQueue;
    ui = !1;
    var a = i.baseQueue
      , o = i.shared.pending;
    if (null !== o) {
        if (null !== a) {
            var u = a.next;
            a.next = o.next,
            o.next = u
        }
        a = o,
        i.shared.pending = null,
        null !== (u = e.alternate) && (null !== (u = u.updateQueue) && (u.baseQueue = o))
    }
    if (null !== a) {
        u = a.next;
        var c = i.baseState
          , s = 0
          , f = null
          , d = null
          , p = null;
        if (null !== u)
            for (var m = u; ; ) {
                if ((o = m.expirationTime) < r) {
                    var h = {
                        expirationTime: m.expirationTime,
                        suspenseConfig: m.suspenseConfig,
                        tag: m.tag,
                        payload: m.payload,
                        callback: m.callback,
                        next: null
                    };
                    null === p ? (d = p = h,
                    f = c) : p = p.next = h,
                    o > s && (s = o)
                } else {
                    null !== p && (p = p.next = {
                        expirationTime: 1073741823,
                        suspenseConfig: m.suspenseConfig,
                        tag: m.tag,
                        payload: m.payload,
                        callback: m.callback,
                        next: null
                    }),
                    pu(o, m.suspenseConfig);
                    e: {
                        var v = e
                          , y = m;
```

```
                                switch (o = t,
                                h = n,
                                y.tag) {
                                case 1:
                                    if ("function" === typeof (v = y.payload)) {
                                        c = v.call(h, c, o);
                                        break e
                                    }
                                    c = v;
                                    break e;
                                case 3:
                                    v.effectTag = -4097 & v.effectTag | 64;
                                case 0:
                                    if (null === (o = "function" === typeof (v = y.payload) ? v.call(h, c, o) : v)
|| void 0 === o)
                                        break e;
                                    c = l({}, c, o);
                                    break e;
                                case 2:
                                    ui = !0
                                }
                            }
                            null !== m.callback && (e.effectTag |= 32,
                            null === (o = i.effects) ? i.effects = [m] : o.push(m))
                        }
                        if (null === (m = m.next) || m === u) {
                            if (null === (o = i.shared.pending))
                                break;
                            m = a.next = o.next,
                            o.next = u,
                            i.baseQueue = a = o,
                            i.shared.pending = null
                        }
                    }
                null === p ? f = c : p.next = d,
                i.baseState = f,
                i.baseQueue = p,
                mu(s),
                e.expirationTime = s,
                e.memoizedState = c
            }
        }
        function hi(e, t, n) {
            if (e = t.effects,
            t.effects = null,
            null !== e)
                for (t = 0; t < e.length; t++) {
                    var r = e[t]
                      , l = r.callback;
                    if (null !== l) {
                        if (r.callback = null,
                        r = l,
                        l = n,
                        "function" !== typeof r)
                            throw Error(a(191, r));
                        r.call(l)
                    }
                }
        }
        var vi = X.ReactCurrentBatchConfig
          , yi = (new r.Component).refs;
        function gi(e, t, n, r) {
            n = null === (n = n(r, t = e.memoizedState)) || void 0 === n ? t : l({}, t, n),
            e.memoizedState = n,
            0 === e.expirationTime && (e.updateQueue.baseState = n)
        }
        var bi = {
            isMounted: function(e) {
                return !!(e = e._reactInternalFiber) && et(e) === e
```

```
            },
            enqueueSetState: function(e, t, n) {
                e = e._reactInternalFiber;
                var r = eu()
                  , l = vi.suspense;
                (l = fi(r = tu(r, e, l), l)).payload = t,
                void 0 !== n && null !== n && (l.callback = n),
                di(e, l),
                nu(e, r)
            },
            enqueueReplaceState: function(e, t, n) {
                e = e._reactInternalFiber;
                var r = eu()
                  , l = vi.suspense;
                (l = fi(r = tu(r, e, l), l)).tag = 1,
                l.payload = t,
                void 0 !== n && null !== n && (l.callback = n),
                di(e, l),
                nu(e, r)
            },
            enqueueForceUpdate: function(e, t) {
                e = e._reactInternalFiber;
                var n = eu()
                  , r = vi.suspense;
                (r = fi(n = tu(n, e, r), r)).tag = 2,
                void 0 !== t && null !== t && (r.callback = t),
                di(e, r),
                nu(e, n)
            }
        };
        function wi(e, t, n, r, l, i, a) {
            return "function" === typeof (e = e.stateNode).shouldComponentUpdate ? e.shouldComponentUpdate(r, i, a)
: !t.prototype || !t.prototype.isPureReactComponent || (!Qr(n, r) || !Qr(l, i))
        }
        function ki(e, t, n) {
            var r = !1
              , l = ml
              , i = t.contextType;
            return "object" === typeof i && null !== i ? i = oi(i) : (l = bl(t) ? yl : hl.current,
            i = (r = null !== (r = t.contextTypes) && void 0 !== r) ? gl(e, l) : ml),
            t = new t(n,i),
            e.memoizedState = null !== t.state && void 0 !== t.state ? t.state : null,
            t.updater = bi,
            e.stateNode = t,
            t._reactInternalFiber = e,
            r && ((e = e.stateNode).__reactInternalMemoizedUnmaskedChildContext = l,
            e.__reactInternalMemoizedMaskedChildContext = i),
            t
        }
        function xi(e, t, n, r) {
            e = t.state,
            "function" === typeof t.componentWillReceiveProps && t.componentWillReceiveProps(n, r),
            "function" === typeof t.UNSAFE_componentWillReceiveProps && t.UNSAFE_componentWillReceiveProps(n, r),
            t.state !== e && bi.enqueueReplaceState(t, t.state, null)
        }
        function Ti(e, t, n, r) {
            var l = e.stateNode;
            l.props = n,
            l.state = e.memoizedState,
            l.refs = yi,
            ci(e);
            var i = t.contextType;
            "object" === typeof i && null !== i ? l.context = oi(i) : (i = bl(t) ? yl : hl.current,
            l.context = gl(e, i)),
            mi(e, n, l, r),
            l.state = e.memoizedState,
            "function" === typeof (i = t.getDerivedStateFromProps) && (gi(e, t, i, n),
            l.state = e.memoizedState),
            "function" === typeof t.getDerivedStateFromProps || "function" === typeof l.getSnapshotBeforeUpdate ||
```

```
"function" !== typeof l.UNSAFE_componentWillMount && "function" !== typeof l.componentWillMount || (t = l.state,
            "function" === typeof l.componentWillMount && l.componentWillMount(),
            "function" === typeof l.UNSAFE_componentWillMount && l.UNSAFE_componentWillMount(),
            t !== l.state && bi.enqueueReplaceState(l, l.state, null),
            mi(e, n, l, r),
            l.state = e.memoizedState),
            "function" === typeof l.componentDidMount && (e.effectTag |= 4)
        }
        var Ei = Array.isArray;
        function Si(e, t, n) {
            if (null !== (e = n.ref) && "function" !== typeof e && "object" !== typeof e) {
                if (n._owner) {
                    if (n = n._owner) {
                        if (1 !== n.tag)
                            throw Error(a(309));
                        var r = n.stateNode
                    }
                    if (!r)
                        throw Error(a(147, e));
                    var l = "" + e;
                    return null !== t && null !== t.ref && "function" === typeof t.ref && t.ref._stringRef === l ?
t.ref : ((t = function(e) {
                        var t = r.refs;
                        t === yi && (t = r.refs = {}),
                        null === e ? delete t[l] : t[l] = e
                    }
                    )._stringRef = l,
                    t)
                }
                if ("string" !== typeof e)
                    throw Error(a(284));
                if (!n._owner)
                    throw Error(a(290, e))
            }
            return e
        }
        function Ci(e, t) {
            if ("textarea" !== e.type)
                throw Error(a(31, "[object Object]" === Object.prototype.toString.call(t) ? "object with keys {" +
Object.keys(t).join(", ") + "}" : t, ""))
        }
        function _i(e) {
            function t(t, n) {
                if (e) {
                    var r = t.lastEffect;
                    null !== r ? (r.nextEffect = n,
                    t.lastEffect = n) : t.firstEffect = t.lastEffect = n,
                    n.nextEffect = null,
                    n.effectTag = 8
                }
            }
            function n(n, r) {
                if (!e)
                    return null;
                for (; null !== r; )
                    t(n, r),
                    r = r.sibling;
                return null
            }
            function r(e, t) {
                for (e = new Map; null !== t; )
                    null !== t.key ? e.set(t.key, t) : e.set(t.index, t),
                    t = t.sibling;
                return e
            }
            function l(e, t) {
                return (e = Mu(e, t)).index = 0,
                e.sibling = null,
                e
```

```
            }
            function i(t, n, r) {
                return t.index = r,
                e ? null !== (r = t.alternate) ? (r = r.index) < n ? (t.effectTag = 2,
                n) : r : (t.effectTag = 2,
                n) : n
            }
            function o(t) {
                return e && null === t.alternate && (t.effectTag = 2),
                t
            }
            function u(e, t, n, r) {
                return null === t || 6 !== t.tag ? ((t = Lu(n, e.mode, r)).return = e,
                t) : ((t = l(t, n)).return = e,
                t)
            }
            function c(e, t, n, r) {
                return null !== t && t.elementType === n.type ? ((r = l(t, n.props)).ref = Si(e, t, n),
                r.return = e,
                r) : ((r = Fu(n.type, n.key, n.props, null, e.mode, r)).ref = Si(e, t, n),
                r.return = e,
                r)
            }
            function s(e, t, n, r) {
                return null === t || 4 !== t.tag || t.stateNode.containerInfo !== n.containerInfo ||
t.stateNode.implementation !== n.implementation ? ((t = Au(n, e.mode, r)).return = e,
                t) : ((t = l(t, n.children || [])).return = e,
                t)
            }
            function f(e, t, n, r, i) {
                return null === t || 7 !== t.tag ? ((t = Du(n, e.mode, r, i)).return = e,
                t) : ((t = l(t, n)).return = e,
                t)
            }
            function d(e, t, n) {
                if ("string" === typeof t || "number" === typeof t)
                    return (t = Lu("" + t, e.mode, n)).return = e,
                    t;
                if ("object" === typeof t && null !== t) {
                    switch (t.$$typeof) {
                    case ee:
                        return (n = Fu(t.type, t.key, t.props, null, e.mode, n)).ref = Si(e, null, t),
                        n.return = e,
                        n;
                    case te:
                        return (t = Au(t, e.mode, n)).return = e,
                        t
                    }
                    if (Ei(t) || he(t))
                        return (t = Du(t, e.mode, n, null)).return = e,
                        t;
                    Ci(e, t)
                }
                return null
            }
            function p(e, t, n, r) {
                var l = null !== t ? t.key : null;
                if ("string" === typeof n || "number" === typeof n)
                    return null !== l ? null : u(e, t, "" + n, r);
                if ("object" === typeof n && null !== n) {
                    switch (n.$$typeof) {
                    case ee:
                        return n.key === l ? n.type === ne ? f(e, t, n.props.children, r, l) : c(e, t, n, r) : null;
                    case te:
                        return n.key === l ? s(e, t, n, r) : null
                    }
                    if (Ei(n) || he(n))
                        return null !== l ? null : f(e, t, n, r, null);
                    Ci(e, n)
```

```
                }
                return null
            }
            function m(e, t, n, r, l) {
                if ("string" === typeof r || "number" === typeof r)
                    return u(t, e = e.get(n) || null, "" + r, l);
                if ("object" === typeof r && null !== r) {
                    switch (r.$$typeof) {
                    case ee:
                        return e = e.get(null === r.key ? n : r.key) || null,
                        r.type === ne ? f(t, e, r.props.children, l, r.key) : c(t, e, r, l);
                    case te:
                        return s(t, e = e.get(null === r.key ? n : r.key) || null, r, l)
                    }
                    if (Ei(r) || he(r))
                        return f(t, e = e.get(n) || null, r, l, null);
                    Ci(t, r)
                }
                return null
            }
            function h(l, a, o, u) {
                for (var c = null, s = null, f = a, h = a = 0, v = null; null !== f && h < o.length; h++) {
                    f.index > h ? (v = f,
                    f = null) : v = f.sibling;
                    var y = p(l, f, o[h], u);
                    if (null === y) {
                        null === f && (f = v);
                        break
                    }
                    e && f && null === y.alternate && t(l, f),
                    a = i(y, a, h),
                    null === s ? c = y : s.sibling = y,
                    s = y,
                    f = v
                }
                if (h === o.length)
                    return n(l, f),
                    c;
                if (null === f) {
                    for (; h < o.length; h++)
                        null !== (f = d(l, o[h], u)) && (a = i(f, a, h),
                        null === s ? c = f : s.sibling = f,
                        s = f);
                    return c
                }
                for (f = r(l, f); h < o.length; h++)
                    null !== (v = m(f, l, h, o[h], u)) && (e && null !== v.alternate && f.delete(null === v.key ? h
: v.key),
                    a = i(v, a, h),
                    null === s ? c = v : s.sibling = v,
                    s = v);
                return e && f.forEach((function(e) {
                    return t(l, e)
                }
                )),
                c
            }
            function v(l, o, u, c) {
                var s = he(u);
                if ("function" !== typeof s)
                    throw Error(a(150));
                if (null == (u = s.call(u)))
                    throw Error(a(151));
                for (var f = s = null, h = o, v = o = 0, y = null, g = u.next(); null !== h && !g.done; v++,
                g = u.next()) {
                    h.index > v ? (y = h,
                    h = null) : y = h.sibling;
                    var b = p(l, h, g.value, c);
                    if (null === b) {
```

```
                        null === h && (h = y);
                        break
                    }
                    e && h && null === b.alternate && t(l, h),
                    o = i(b, o, v),
                    null === f ? s = b : f.sibling = b,
                    f = b,
                    h = y
                }
                if (g.done)
                    return n(l, h),
                    s;
                if (null === h) {
                    for (; !g.done; v++,
                    g = u.next())
                        null !== (g = d(l, g.value, c)) && (o = i(g, o, v),
                        null === f ? s = g : f.sibling = g,
                        f = g);
                    return s
                }
                for (h = r(l, h); !g.done; v++,
                g = u.next())
                    null !== (g = m(h, l, v, g.value, c)) && (e && null !== g.alternate && h.delete(null === g.key ?
v : g.key),
                    o = i(g, o, v),
                    null === f ? s = g : f.sibling = g,
                    f = g);
                return e && h.forEach((function(e) {
                    return t(l, e)
                }
                )),
                s
            }
            return function(e, r, i, u) {
                var c = "object" === typeof i && null !== i && i.type === ne && null === i.key;
                c && (i = i.props.children);
                var s = "object" === typeof i && null !== i;
                if (s)
                    switch (i.$$typeof) {
                    case ee:
                        e: {
                            for (s = i.key,
                            c = r; null !== c; ) {
                                if (c.key === s) {
                                    switch (c.tag) {
                                    case 7:
                                        if (i.type === ne) {
                                            n(e, c.sibling),
                                            (r = l(c, i.props.children)).return = e,
                                            e = r;
                                            break e
                                        }
                                        break;
                                    default:
                                        if (c.elementType === i.type) {
                                            n(e, c.sibling),
                                            (r = l(c, i.props)).ref = Si(e, c, i),
                                            r.return = e,
                                            e = r;
                                            break e
                                        }
                                    }
                                    n(e, c);
                                    break
                                }
                                t(e, c),
                                c = c.sibling
                            }
                            i.type === ne ? ((r = Du(i.props.children, e.mode, u, i.key)).return = e,
```

```
                            e = r) : ((u = Fu(i.type, i.key, i.props, null, e.mode, u)).ref = Si(e, r, i),
                            u.return = e,
                            e = u)
                        }
                        return o(e);
                    case te:
                        e: {
                            for (c = i.key; null !== r; ) {
                                if (r.key === c) {
                                    if (4 === r.tag && r.stateNode.containerInfo === i.containerInfo &&
r.stateNode.implementation === i.implementation) {
                                        n(e, r.sibling),
                                        (r = l(r, i.children || [])).return = e,
                                        e = r;
                                        break e
                                    }
                                    n(e, r);
                                    break
                                }
                                t(e, r),
                                r = r.sibling
                            }
                            (r = Au(i, e.mode, u)).return = e,
                            e = r
                        }
                        return o(e)
                    }
                if ("string" === typeof i || "number" === typeof i)
                    return i = "" + i,
                    null !== r && 6 === r.tag ? (n(e, r.sibling),
                    (r = l(r, i)).return = e,
                    e = r) : (n(e, r),
                    (r = Lu(i, e.mode, u)).return = e,
                    e = r),
                    o(e);
                if (Ei(i))
                    return h(e, r, i, u);
                if (he(i))
                    return v(e, r, i, u);
                if (s && Ci(e, i),
                "undefined" === typeof i && !c)
                    switch (e.tag) {
                    case 1:
                    case 0:
                        throw e = e.type,
                        Error(a(152, e.displayName || e.name || "Component"))
                    }
                return n(e, r)
            }
        }
        var Pi = _i(!0)
          , Ni = _i(!1)
          , zi = {}
          , Oi = {
            current: zi
        }
          , Ri = {
            current: zi
        }
          , Ii = {
            current: zi
        };
        function Mi(e) {
            if (e === zi)
                throw Error(a(174));
            return e
        }
        function Fi(e, t) {
            switch (pl(Ii, t),
```

```
    pl(Ri, e),
    pl(Oi, zi),
    e = t.nodeType) {
    case 9:
    case 11:
        t = (t = t.documentElement) ? t.namespaceURI : Le(null, "");
        break;
    default:
        t = Le(t = (e = 8 === e ? t.parentNode : t).namespaceURI || null, e = e.tagName)
    }
    dl(Oi),
    pl(Oi, t)
}
function Di() {
    dl(Oi),
    dl(Ri),
    dl(Ii)
}
function Li(e) {
    Mi(Ii.current);
    var t = Mi(Oi.current)
      , n = Le(t, e.type);
    t !== n && (pl(Ri, e),
    pl(Oi, n))
}
function Ai(e) {
    Ri.current === e && (dl(Oi),
    dl(Ri))
}
var Ui = {
    current: 0
};
function Vi(e) {
    for (var t = e; null !== t; ) {
        if (13 === t.tag) {
            var n = t.memoizedState;
            if (null !== n && (null === (n = n.dehydrated) || n.data === yn || n.data === gn))
                return t
        } else if (19 === t.tag && void 0 !== t.memoizedProps.revealOrder) {
            if (0 !== (64 & t.effectTag))
                return t
        } else if (null !== t.child) {
            t.child.return = t,
            t = t.child;
            continue
        }
        if (t === e)
            break;
        for (; null === t.sibling; ) {
            if (null === t.return || t.return === e)
                return null;
            t = t.return
        }
        t.sibling.return = t.return,
        t = t.sibling
    }
    return null
}
function Wi(e, t) {
    return {
        responder: e,
        props: t
    }
}
var ji = X.ReactCurrentDispatcher
  , Qi = X.ReactCurrentBatchConfig
  , $i = 0
  , Bi = null
  , Hi = null
```

```
  , Ki = null
  , qi = !1;
function Yi() {
    throw Error(a(321))
}
function Xi(e, t) {
    if (null === t)
        return !1;
    for (var n = 0; n < t.length && n < e.length; n++)
        if (!Wr(e[n], t[n]))
            return !1;
    return !0
}
function Gi(e, t, n, r, l, i) {
    if ($i = i,
    Bi = t,
    t.memoizedState = null,
    t.updateQueue = null,
    t.expirationTime = 0,
    ji.current = null === e || null === e.memoizedState ? wa : ka,
    e = n(r, l),
    t.expirationTime === $i) {
        i = 0;
        do {
            if (t.expirationTime = 0,
            !(25 > i))
                throw Error(a(301));
            i += 1,
            Ki = Hi = null,
            t.updateQueue = null,
            ji.current = xa,
            e = n(r, l)
        } while (t.expirationTime === $i)
    }
    if (ji.current = ba,
    t = null !== Hi && null !== Hi.next,
    $i = 0,
    Ki = Hi = Bi = null,
    qi = !1,
    t)
        throw Error(a(300));
    return e
}
function Zi() {
    var e = {
        memoizedState: null,
        baseState: null,
        baseQueue: null,
        queue: null,
        next: null
    };
    return null === Ki ? Bi.memoizedState = Ki = e : Ki = Ki.next = e,
    Ki
}
function Ji() {
    if (null === Hi) {
        var e = Bi.alternate;
        e = null !== e ? e.memoizedState : null
    } else
        e = Hi.next;
    var t = null === Ki ? Bi.memoizedState : Ki.next;
    if (null !== t)
        Ki = t,
        Hi = e;
    else {
        if (null === e)
            throw Error(a(310));
        e = {
            memoizedState: (Hi = e).memoizedState,
```

```
            baseState: Hi.baseState,
            baseQueue: Hi.baseQueue,
            queue: Hi.queue,
            next: null
        },
        null === Ki ? Bi.memoizedState = Ki = e : Ki = Ki.next = e
    }
    return Ki
}
function ea(e, t) {
    return "function" === typeof t ? t(e) : t
}
function ta(e) {
    var t = Ji()
      , n = t.queue;
    if (null === n)
        throw Error(a(311));
    n.lastRenderedReducer = e;
    var r = Hi
      , l = r.baseQueue
      , i = n.pending;
    if (null !== i) {
        if (null !== l) {
            var o = l.next;
            l.next = i.next,
            i.next = o
        }
        r.baseQueue = l = i,
        n.pending = null
    }
    if (null !== l) {
        l = l.next,
        r = r.baseState;
        var u = o = i = null
          , c = l;
        do {
            var s = c.expirationTime;
            if (s < $i) {
                var f = {
                    expirationTime: c.expirationTime,
                    suspenseConfig: c.suspenseConfig,
                    action: c.action,
                    eagerReducer: c.eagerReducer,
                    eagerState: c.eagerState,
                    next: null
                };
                null === u ? (o = u = f,
                i = r) : u = u.next = f,
                s > Bi.expirationTime && (Bi.expirationTime = s,
                mu(s))
            } else
                null !== u && (u = u.next = {
                    expirationTime: 1073741823,
                    suspenseConfig: c.suspenseConfig,
                    action: c.action,
                    eagerReducer: c.eagerReducer,
                    eagerState: c.eagerState,
                    next: null
                }),
                pu(s, c.suspenseConfig),
                r = c.eagerReducer === e ? c.eagerState : e(r, c.action);
            c = c.next
        } while (null !== c && c !== l);
        null === u ? i = r : u.next = o,
        Wr(r, t.memoizedState) || (Ia = !0),
        t.memoizedState = r,
        t.baseState = i,
        t.baseQueue = u,
        n.lastRenderedState = r
```

```
    }
    return [t.memoizedState, n.dispatch]
}
function na(e) {
    var t = Ji()
      , n = t.queue;
    if (null === n)
        throw Error(a(311));
    n.lastRenderedReducer = e;
    var r = n.dispatch
      , l = n.pending
      , i = t.memoizedState;
    if (null !== l) {
        n.pending = null;
        var o = l = l.next;
        do {
            i = e(i, o.action),
            o = o.next
        } while (o !== l);
        Wr(i, t.memoizedState) || (Ia = !0),
        t.memoizedState = i,
        null === t.baseQueue && (t.baseState = i),
        n.lastRenderedState = i
    }
    return [i, r]
}
function ra(e) {
    var t = Zi();
    return "function" === typeof e && (e = e()),
    t.memoizedState = t.baseState = e,
    e = (e = t.queue = {
        pending: null,
        dispatch: null,
        lastRenderedReducer: ea,
        lastRenderedState: e
    }).dispatch = ga.bind(null, Bi, e),
    [t.memoizedState, e]
}
function la(e, t, n, r) {
    return e = {
        tag: e,
        create: t,
        destroy: n,
        deps: r,
        next: null
    },
    null === (t = Bi.updateQueue) ? (t = {
        lastEffect: null
    },
    Bi.updateQueue = t,
    t.lastEffect = e.next = e) : null === (n = t.lastEffect) ? t.lastEffect = e.next = e : (r = n.next,
    n.next = e,
    e.next = r,
    t.lastEffect = e),
    e
}
function ia() {
    return Ji().memoizedState
}
function aa(e, t, n, r) {
    var l = Zi();
    Bi.effectTag |= e,
    l.memoizedState = la(1 | t, n, void 0, void 0 === r ? null : r)
}
function oa(e, t, n, r) {
    var l = Ji();
    r = void 0 === r ? null : r;
    var i = void 0;
    if (null !== Hi) {
```

```
        var a = Hi.memoizedState;
        if (i = a.destroy,
        null !== r && Xi(r, a.deps))
            return void la(t, n, i, r)
    }
    Bi.effectTag |= e,
    l.memoizedState = la(1 | t, n, i, r)
}
function ua(e, t) {
    return aa(516, 4, e, t)
}
function ca(e, t) {
    return oa(516, 4, e, t)
}
function sa(e, t) {
    return oa(4, 2, e, t)
}
function fa(e, t) {
    return "function" === typeof t ? (e = e(),
    t(e),
    function() {
        t(null)
    }
    ) : null !== t && void 0 !== t ? (e = e(),
    t.current = e,
    function() {
        t.current = null
    }
    ) : void 0
}
function da(e, t, n) {
    return n = null !== n && void 0 !== n ? n.concat([e]) : null,
    oa(4, 2, fa.bind(null, t, e), n)
}
function pa() {}
function ma(e, t) {
    return Zi().memoizedState = [e, void 0 === t ? null : t],
    e
}
function ha(e, t) {
    var n = Ji();
    t = void 0 === t ? null : t;
    var r = n.memoizedState;
    return null !== r && null !== t && Xi(t, r[1]) ? r[0] : (n.memoizedState = [e, t],
    e)
}
function va(e, t) {
    var n = Ji();
    t = void 0 === t ? null : t;
    var r = n.memoizedState;
    return null !== r && null !== t && Xi(t, r[1]) ? r[0] : (e = e(),
    n.memoizedState = [e, t],
    e)
}
function ya(e, t, n) {
    var r = $l();
    Hl(98 > r ? 98 : r, (function() {
        e(!0)
    }
    )),
    Hl(97 < r ? 97 : r, (function() {
        var r = Qi.suspense;
        Qi.suspense = void 0 === t ? null : t;
        try {
            e(!1),
            n()
        } finally {
            Qi.suspense = r
        }
```

```
            }
            ))
        }
        function ga(e, t, n) {
            var r = eu()
              , l = vi.suspense;
            l = {
                expirationTime: r = tu(r, e, l),
                suspenseConfig: l,
                action: n,
                eagerReducer: null,
                eagerState: null,
                next: null
            };
            var i = t.pending;
            if (null === i ? l.next = l : (l.next = i.next,
            i.next = l),
            t.pending = l,
            i = e.alternate,
            e === Bi || null !== i && i === Bi)
                qi = !0,
                l.expirationTime = $i,
                Bi.expirationTime = $i;
            else {
                if (0 === e.expirationTime && (null === i || 0 === i.expirationTime) && null !== (i =
t.lastRenderedReducer))
                    try {
                        var a = t.lastRenderedState
                          , o = i(a, n);
                        if (l.eagerReducer = i,
                        l.eagerState = o,
                        Wr(o, a))
                            return
                    } catch (u) {}
                nu(e, r)
            }
        }
        var ba = {
            readContext: oi,
            useCallback: Yi,
            useContext: Yi,
            useEffect: Yi,
            useImperativeHandle: Yi,
            useLayoutEffect: Yi,
            useMemo: Yi,
            useReducer: Yi,
            useRef: Yi,
            useState: Yi,
            useDebugValue: Yi,
            useResponder: Yi,
            useDeferredValue: Yi,
            useTransition: Yi
        }
          , wa = {
            readContext: oi,
            useCallback: ma,
            useContext: oi,
            useEffect: ua,
            useImperativeHandle: function(e, t, n) {
                return n = null !== n && void 0 !== n ? n.concat([e]) : null,
                aa(4, 2, fa.bind(null, t, e), n)
            },
            useLayoutEffect: function(e, t) {
                return aa(4, 2, e, t)
            },
            useMemo: function(e, t) {
                var n = Zi();
                return t = void 0 === t ? null : t,
                e = e(),
```

```
        n.memoizedState = [e, t],
        e
    },
    useReducer: function(e, t, n) {
        var r = Zi();
        return t = void 0 !== n ? n(t) : t,
        r.memoizedState = r.baseState = t,
        e = (e = r.queue = {
            pending: null,
            dispatch: null,
            lastRenderedReducer: e,
            lastRenderedState: t
        }).dispatch = ga.bind(null, Bi, e),
        [r.memoizedState, e]
    },
    useRef: function(e) {
        return e = {
            current: e
        },
        Zi().memoizedState = e
    },
    useState: ra,
    useDebugValue: pa,
    useResponder: Wi,
    useDeferredValue: function(e, t) {
        var n = ra(e)
          , r = n[0]
          , l = n[1];
        return ua((function() {
            var n = Qi.suspense;
            Qi.suspense = void 0 === t ? null : t;
            try {
                l(e)
            } finally {
                Qi.suspense = n
            }
        }
        ), [e, t]),
        r
    },
    useTransition: function(e) {
        var t = ra(!1)
          , n = t[0];
        return t = t[1],
        [ma(ya.bind(null, t, e), [t, e]), n]
    }
}
  , ka = {
    readContext: oi,
    useCallback: ha,
    useContext: oi,
    useEffect: ca,
    useImperativeHandle: da,
    useLayoutEffect: sa,
    useMemo: va,
    useReducer: ta,
    useRef: ia,
    useState: function() {
        return ta(ea)
    },
    useDebugValue: pa,
    useResponder: Wi,
    useDeferredValue: function(e, t) {
        var n = ta(ea)
          , r = n[0]
          , l = n[1];
        return ca((function() {
            var n = Qi.suspense;
            Qi.suspense = void 0 === t ? null : t;
```

```
            try {
                l(e)
            } finally {
                Qi.suspense = n
            }
        }
        ), [e, t]),
        r
    },
    useTransition: function(e) {
        var t = ta(ea)
          , n = t[0];
        return t = t[1],
        [ha(ya.bind(null, t, e), [t, e]), n]
    }
}
  , xa = {
    readContext: oi,
    useCallback: ha,
    useContext: oi,
    useEffect: ca,
    useImperativeHandle: da,
    useLayoutEffect: sa,
    useMemo: va,
    useReducer: na,
    useRef: ia,
    useState: function() {
        return na(ea)
    },
    useDebugValue: pa,
    useResponder: Wi,
    useDeferredValue: function(e, t) {
        var n = na(ea)
          , r = n[0]
          , l = n[1];
        return ca((function() {
            var n = Qi.suspense;
            Qi.suspense = void 0 === t ? null : t;
            try {
                l(e)
            } finally {
                Qi.suspense = n
            }
        }
        ), [e, t]),
        r
    },
    useTransition: function(e) {
        var t = na(ea)
          , n = t[0];
        return t = t[1],
        [ha(ya.bind(null, t, e), [t, e]), n]
    }
}
  , Ta = null
  , Ea = null
  , Sa = !1;
function Ca(e, t) {
    var n = Ru(5, null, null, 0);
    n.elementType = "DELETED",
    n.type = "DELETED",
    n.stateNode = t,
    n.return = e,
    n.effectTag = 8,
    null !== e.lastEffect ? (e.lastEffect.nextEffect = n,
    e.lastEffect = n) : e.firstEffect = e.lastEffect = n
}
function _a(e, t) {
    switch (e.tag) {
```

```
            case 5:
                var n = e.type;
                return null !== (t = 1 !== t.nodeType || n.toLowerCase() !== t.nodeName.toLowerCase() ? null : t) &&
(e.stateNode = t,
                !0);
            case 6:
                return null !== (t = "" === e.pendingProps || 3 !== t.nodeType ? null : t) && (e.stateNode = t,
                !0);
            case 13:
            default:
                return !1
            }
        }
        function Pa(e) {
            if (Sa) {
                var t = Ea;
                if (t) {
                    var n = t;
                    if (!_a(e, t)) {
                        if (!(t = Sn(n.nextSibling)) || !_a(e, t))
                            return e.effectTag = -1025 & e.effectTag | 2,
                            Sa = !1,
                            void (Ta = e);
                        Ca(Ta, n)
                    }
                    Ta = e,
                    Ea = Sn(t.firstChild)
                } else
                    e.effectTag = -1025 & e.effectTag | 2,
                    Sa = !1,
                    Ta = e
            }
        }
        function Na(e) {
            for (e = e.return; null !== e && 5 !== e.tag && 3 !== e.tag && 13 !== e.tag; )
                e = e.return;
            Ta = e
        }
        function za(e) {
            if (e !== Ta)
                return !1;
            if (!Sa)
                return Na(e),
                Sa = !0,
                !1;
            var t = e.type;
            if (5 !== e.tag || "head" !== t && "body" !== t && !xn(t, e.memoizedProps))
                for (t = Ea; t; )
                    Ca(e, t),
                    t = Sn(t.nextSibling);
            if (Na(e),
            13 === e.tag) {
                if (!(e = null !== (e = e.memoizedState) ? e.dehydrated : null))
                    throw Error(a(317));
                e: {
                    for (e = e.nextSibling,
                    t = 0; e; ) {
                        if (8 === e.nodeType) {
                            var n = e.data;
                            if ("/$" === n) {
                                if (0 === t) {
                                    Ea = Sn(e.nextSibling);
                                    break e
                                }
                                t--
                            } else
                                "$" !== n && n !== gn && n !== yn || t++
                        }
                        e = e.nextSibling
```

```
                }
                Ea = null
            }
        } else
            Ea = Ta ? Sn(e.stateNode.nextSibling) : null;
        return !0
    }
    function Oa() {
        Ea = Ta = null,
        Sa = !1
    }
    var Ra = X.ReactCurrentOwner
      , Ia = !1;
    function Ma(e, t, n, r) {
        t.child = null === e ? Ni(t, null, n, r) : Pi(t, e.child, n, r)
    }
    function Fa(e, t, n, r, l) {
        n = n.render;
        var i = t.ref;
        return ai(t, l),
        r = Gi(e, t, n, r, i, l),
        null === e || Ia ? (t.effectTag |= 1,
        Ma(e, t, r, l),
        t.child) : (t.updateQueue = e.updateQueue,
        t.effectTag &= -517,
        e.expirationTime <= l && (e.expirationTime = 0),
        Ga(e, t, l))
    }
    function Da(e, t, n, r, l, i) {
        if (null === e) {
            var a = n.type;
            return "function" !== typeof a || Iu(a) || void 0 !== a.defaultProps || null !== n.compare || void 0
!== n.defaultProps ? ((e = Fu(n.type, null, r, null, t.mode, i)).ref = t.ref,
            e.return = t,
            t.child = e) : (t.tag = 15,
            t.type = a,
            La(e, t, a, r, l, i))
        }
        return a = e.child,
        l < i && (l = a.memoizedProps,
        (n = null !== (n = n.compare) ? n : Qr)(l, r) && e.ref === t.ref) ? Ga(e, t, i) : (t.effectTag |= 1,
        (e = Mu(a, r)).ref = t.ref,
        e.return = t,
        t.child = e)
    }
    function La(e, t, n, r, l, i) {
        return null !== e && Qr(e.memoizedProps, r) && e.ref === t.ref && (Ia = !1,
        l < i) ? (t.expirationTime = e.expirationTime,
        Ga(e, t, i)) : Ua(e, t, n, r, i)
    }
    function Aa(e, t) {
        var n = t.ref;
        (null === e && null !== n || null !== e && e.ref !== n) && (t.effectTag |= 128)
    }
    function Ua(e, t, n, r, l) {
        var i = bl(n) ? yl : hl.current;
        return i = gl(t, i),
        ai(t, l),
        n = Gi(e, t, n, r, i, l),
        null === e || Ia ? (t.effectTag |= 1,
        Ma(e, t, n, l),
        t.child) : (t.updateQueue = e.updateQueue,
        t.effectTag &= -517,
        e.expirationTime <= l && (e.expirationTime = 0),
        Ga(e, t, l))
    }
    function Va(e, t, n, r, l) {
        if (bl(n)) {
            var i = !0;
```

```
                Tl(t)
            } else
                i = !1;
            if (ai(t, l),
            null === t.stateNode)
                null !== e && (e.alternate = null,
                t.alternate = null,
                t.effectTag |= 2),
                ki(t, n, r),
                Ti(t, n, r, l),
                r = !0;
            else if (null === e) {
                var a = t.stateNode
                  , o = t.memoizedProps;
                a.props = o;
                var u = a.context
                  , c = n.contextType;
                "object" === typeof c && null !== c ? c = oi(c) : c = gl(t, c = bl(n) ? yl : hl.current);
                var s = n.getDerivedStateFromProps
                  , f = "function" === typeof s || "function" === typeof a.getSnapshotBeforeUpdate;
                f || "function" !== typeof a.UNSAFE_componentWillReceiveProps && "function" !== typeof
a.componentWillReceiveProps || (o !== r || u !== c) && xi(t, a, r, c),
                ui = !1;
                var d = t.memoizedState;
                a.state = d,
                mi(t, r, a, l),
                u = t.memoizedState,
                o !== r || d !== u || vl.current || ui ? ("function" === typeof s && (gi(t, n, s, r),
                u = t.memoizedState),
                (o = ui || wi(t, n, o, r, d, u, c)) ? (f || "function" !== typeof a.UNSAFE_componentWillMount &&
"function" !== typeof a.componentWillMount || ("function" === typeof a.componentWillMount && a.componentWillMount(),
                "function" === typeof a.UNSAFE_componentWillMount && a.UNSAFE_componentWillMount()),
                "function" === typeof a.componentDidMount && (t.effectTag |= 4)) : ("function" === typeof
a.componentDidMount && (t.effectTag |= 4),
                t.memoizedProps = r,
                t.memoizedState = u),
                a.props = r,
                a.state = u,
                a.context = c,
                r = o) : ("function" === typeof a.componentDidMount && (t.effectTag |= 4),
                r = !1)
            } else
                a = t.stateNode,
                si(e, t),
                o = t.memoizedProps,
                a.props = t.type === t.elementType ? o : Zl(t.type, o),
                u = a.context,
                "object" === typeof (c = n.contextType) && null !== c ? c = oi(c) : c = gl(t, c = bl(n) ? yl :
hl.current),
                (f = "function" === typeof (s = n.getDerivedStateFromProps) || "function" === typeof
a.getSnapshotBeforeUpdate) || "function" !== typeof a.UNSAFE_componentWillReceiveProps && "function" !== typeof
a.componentWillReceiveProps || (o !== r || u !== c) && xi(t, a, r, c),
                ui = !1,
                u = t.memoizedState,
                a.state = u,
                mi(t, r, a, l),
                d = t.memoizedState,
                o !== r || u !== d || vl.current || ui ? ("function" === typeof s && (gi(t, n, s, r),
                d = t.memoizedState),
                (s = ui || wi(t, n, o, r, u, d, c)) ? (f || "function" !== typeof a.UNSAFE_componentWillUpdate &&
"function" !== typeof a.componentWillUpdate || ("function" === typeof a.componentWillUpdate &&
a.componentWillUpdate(r, d, c),
                "function" === typeof a.UNSAFE_componentWillUpdate && a.UNSAFE_componentWillUpdate(r, d, c)),
                "function" === typeof a.componentDidUpdate && (t.effectTag |= 4),
                "function" === typeof a.getSnapshotBeforeUpdate && (t.effectTag |= 256)) : ("function" !== typeof
a.componentDidUpdate || o === e.memoizedProps && u === e.memoizedState || (t.effectTag |= 4),
                "function" !== typeof a.getSnapshotBeforeUpdate || o === e.memoizedProps && u === e.memoizedState ||
(t.effectTag |= 256),
                t.memoizedProps = r,
```

```
                t.memoizedState = d),
                a.props = r,
                a.state = d,
                a.context = c,
                r = s) : ("function" !== typeof a.componentDidUpdate || o === e.memoizedProps && u ===
e.memoizedState || (t.effectTag |= 4),
                "function" !== typeof a.getSnapshotBeforeUpdate || o === e.memoizedProps && u === e.memoizedState ||
(t.effectTag |= 256),
                r = !1);
            return Wa(e, t, n, r, i, l)
        }
        function Wa(e, t, n, r, l, i) {
            Aa(e, t);
            var a = 0 !== (64 & t.effectTag);
            if (!r && !a)
                return l && El(t, n, !1),
                Ga(e, t, i);
            r = t.stateNode,
            Ra.current = t;
            var o = a && "function" !== typeof n.getDerivedStateFromError ? null : r.render();
            return t.effectTag |= 1,
            null !== e && a ? (t.child = Pi(t, e.child, null, i),
            t.child = Pi(t, null, o, i)) : Ma(e, t, o, i),
            t.memoizedState = r.state,
            l && El(t, n, !0),
            t.child
        }
        function ja(e) {
            var t = e.stateNode;
            t.pendingContext ? kl(0, t.pendingContext, t.pendingContext !== t.context) : t.context && kl(0,
t.context, !1),
            Fi(e, t.containerInfo)
        }
        var Qa, $a, Ba, Ha = {
            dehydrated: null,
            retryTime: 0
        };
        function Ka(e, t, n) {
            var r, l = t.mode, i = t.pendingProps, a = Ui.current, o = !1;
            if ((r = 0 !== (64 & t.effectTag)) || (r = 0 !== (2 & a) && (null === e || null !== e.memoizedState)),
            r ? (o = !0,
            t.effectTag &= -65) : null !== e && null === e.memoizedState || void 0 === i.fallback || !0 ===
i.unstable_avoidThisFallback || (a |= 1),
            pl(Ui, 1 & a),
            null === e) {
                if (void 0 !== i.fallback && Pa(t),
                o) {
                    if (o = i.fallback,
                    (i = Du(null, l, 0, null)).return = t,
                    0 === (2 & t.mode))
                        for (e = null !== t.memoizedState ? t.child.child : t.child,
                        i.child = e; null !== e; )
                            e.return = i,
                            e = e.sibling;
                    return (n = Du(o, l, n, null)).return = t,
                    i.sibling = n,
                    t.memoizedState = Ha,
                    t.child = i,
                    n
                }
                return l = i.children,
                t.memoizedState = null,
                t.child = Ni(t, null, l, n)
            }
            if (null !== e.memoizedState) {
                if (l = (e = e.child).sibling,
                o) {
                    if (i = i.fallback,
                    (n = Mu(e, e.pendingProps)).return = t,
```

```
            0 === (2 & t.mode) && (o = null !== t.memoizedState ? t.child.child : t.child) !== e.child)
                for (n.child = o; null !== o; )
                    o.return = n,
                    o = o.sibling;
            return (l = Mu(l, i)).return = t,
            n.sibling = l,
            n.childExpirationTime = 0,
            t.memoizedState = Ha,
            t.child = n,
            l
        }
        return n = Pi(t, e.child, i.children, n),
        t.memoizedState = null,
        t.child = n
    }
    if (e = e.child,
    o) {
        if (o = i.fallback,
        (i = Du(null, l, 0, null)).return = t,
        i.child = e,
        null !== e && (e.return = i),
        0 === (2 & t.mode))
            for (e = null !== t.memoizedState ? t.child.child : t.child,
            i.child = e; null !== e; )
                e.return = i,
                e = e.sibling;
        return (n = Du(o, l, n, null)).return = t,
        i.sibling = n,
        n.effectTag |= 2,
        i.childExpirationTime = 0,
        t.memoizedState = Ha,
        t.child = i,
        n
    }
    return t.memoizedState = null,
    t.child = Pi(t, e, i.children, n)
}
function qa(e, t) {
    e.expirationTime < t && (e.expirationTime = t);
    var n = e.alternate;
    null !== n && n.expirationTime < t && (n.expirationTime = t),
    ii(e.return, t)
}
function Ya(e, t, n, r, l, i) {
    var a = e.memoizedState;
    null === a ? e.memoizedState = {
        isBackwards: t,
        rendering: null,
        renderingStartTime: 0,
        last: r,
        tail: n,
        tailExpiration: 0,
        tailMode: l,
        lastEffect: i
    } : (a.isBackwards = t,
    a.rendering = null,
    a.renderingStartTime = 0,
    a.last = r,
    a.tail = n,
    a.tailExpiration = 0,
    a.tailMode = l,
    a.lastEffect = i)
}
function Xa(e, t, n) {
    var r = t.pendingProps
      , l = r.revealOrder
      , i = r.tail;
    if (Ma(e, t, r.children, n),
    0 !== (2 & (r = Ui.current)))
```

```
        r = 1 & r | 2,
        t.effectTag |= 64;
    else {
        if (null !== e && 0 !== (64 & e.effectTag))
            e: for (e = t.child; null !== e; ) {
                if (13 === e.tag)
                    null !== e.memoizedState && qa(e, n);
                else if (19 === e.tag)
                    qa(e, n);
                else if (null !== e.child) {
                    e.child.return = e,
                    e = e.child;
                    continue
                }
                if (e === t)
                    break e;
                for (; null === e.sibling; ) {
                    if (null === e.return || e.return === t)
                        break e;
                    e = e.return
                }
                e.sibling.return = e.return,
                e = e.sibling
            }
        r &= 1
    }
    if (pl(Ui, r),
    0 === (2 & t.mode))
        t.memoizedState = null;
    else
        switch (l) {
        case "forwards":
            for (n = t.child,
            l = null; null !== n; )
                null !== (e = n.alternate) && null === Vi(e) && (l = n),
                n = n.sibling;
            null === (n = l) ? (l = t.child,
            t.child = null) : (l = n.sibling,
            n.sibling = null),
            Ya(t, !1, l, n, i, t.lastEffect);
            break;
        case "backwards":
            for (n = null,
            l = t.child,
            t.child = null; null !== l; ) {
                if (null !== (e = l.alternate) && null === Vi(e)) {
                    t.child = l;
                    break
                }
                e = l.sibling,
                l.sibling = n,
                n = l,
                l = e
            }
            Ya(t, !0, n, null, i, t.lastEffect);
            break;
        case "together":
            Ya(t, !1, null, null, void 0, t.lastEffect);
            break;
        default:
            t.memoizedState = null
        }
    return t.child
}
function Ga(e, t, n) {
    null !== e && (t.dependencies = e.dependencies);
    var r = t.expirationTime;
    if (0 !== r && mu(r),
    t.childExpirationTime < n)
```

```
        return null;
    if (null !== e && t.child !== e.child)
        throw Error(a(153));
    if (null !== t.child) {
        for (n = Mu(e = t.child, e.pendingProps),
        t.child = n,
        n.return = t; null !== e.sibling; )
            e = e.sibling,
            (n = n.sibling = Mu(e, e.pendingProps)).return = t;
        n.sibling = null
    }
    return t.child
}
function Za(e, t) {
    switch (e.tailMode) {
    case "hidden":
        t = e.tail;
        for (var n = null; null !== t; )
            null !== t.alternate && (n = t),
            t = t.sibling;
        null === n ? e.tail = null : n.sibling = null;
        break;
    case "collapsed":
        n = e.tail;
        for (var r = null; null !== n; )
            null !== n.alternate && (r = n),
            n = n.sibling;
        null === r ? t || null === e.tail ? e.tail = null : e.tail.sibling = null : r.sibling = null
    }
}
function Ja(e, t, n) {
    var r = t.pendingProps;
    switch (t.tag) {
    case 2:
    case 16:
    case 15:
    case 0:
    case 11:
    case 7:
    case 8:
    case 12:
    case 9:
    case 14:
        return null;
    case 1:
        return bl(t.type) && wl(),
        null;
    case 3:
        return Di(),
        dl(vl),
        dl(hl),
        (n = t.stateNode).pendingContext && (n.context = n.pendingContext,
        n.pendingContext = null),
        null !== e && null !== e.child || !za(t) || (t.effectTag |= 4),
        null;
    case 5:
        Ai(t),
        n = Mi(Ii.current);
        var i = t.type;
        if (null !== e && null != t.stateNode)
            $a(e, t, i, r, n),
            e.ref !== t.ref && (t.effectTag |= 128);
        else {
            if (!r) {
                if (null === t.stateNode)
                    throw Error(a(166));
                return null
            }
            if (e = Mi(Oi.current),
```

```
                    za(t)) {
                        r = t.stateNode,
                        i = t.type;
                        var o = t.memoizedProps;
                        switch (r[Pn] = t,
                        r[Nn] = o,
                        i) {
                        case "iframe":
                        case "object":
                        case "embed":
                            qt("load", r);
                            break;
                        case "video":
                        case "audio":
                            for (e = 0; e < Ge.length; e++)
                                qt(Ge[e], r);
                            break;
                        case "source":
                            qt("error", r);
                            break;
                        case "img":
                        case "image":
                        case "link":
                            qt("error", r),
                            qt("load", r);
                            break;
                        case "form":
                            qt("reset", r),
                            qt("submit", r);
                            break;
                        case "details":
                            qt("toggle", r);
                            break;
                        case "input":
                            Te(r, o),
                            qt("invalid", r),
                            cn(n, "onChange");
                            break;
                        case "select":
                            r._wrapperState = {
                                wasMultiple: !!o.multiple
                            },
                            qt("invalid", r),
                            cn(n, "onChange");
                            break;
                        case "textarea":
                            Oe(r, o),
                            qt("invalid", r),
                            cn(n, "onChange")
                        }
                        for (var u in an(i, o),
                        e = null,
                        o)
                            if (o.hasOwnProperty(u)) {
                                var c = o[u];
                                "children" === u ? "string" === typeof c ? r.textContent !== c && (e = ["children",
c]) : "number" === typeof c && r.textContent !== "" + c && (e = ["children", "" + c]) : E.hasOwnProperty(u) && null
!= c && cn(n, u)
                            }
                        switch (i) {
                        case "input":
                            we(r),
                            Ce(r, o, !0);
                            break;
                        case "textarea":
                            we(r),
                            Ie(r);
                            break;
                        case "select":
```

```
    case "option":
        break;
    default:
        "function" === typeof o.onClick && (r.onclick = sn)
    }
    n = e,
    t.updateQueue = n,
    null !== n && (t.effectTag |= 4)
} else {
    switch (u = 9 === n.nodeType ? n : n.ownerDocument,
    e === un && (e = De(i)),
    e === un ? "script" === i ? ((e = u.createElement("div")).innerHTML = "<script><\/script>",
    e = e.removeChild(e.firstChild)) : "string" === typeof r.is ? e = u.createElement(i, {
        is: r.is
    }) : (e = u.createElement(i),
    "select" === i && (u = e,
    r.multiple ? u.multiple = !0 : r.size && (u.size = r.size))) : e = u.createElementNS(e, i),
    e[Pn] = t,
    e[Nn] = r,
    Qa(e, t),
    t.stateNode = e,
    u = on(i, r),
    i) {
    case "iframe":
    case "object":
    case "embed":
        qt("load", e),
        c = r;
        break;
    case "video":
    case "audio":
        for (c = 0; c < Ge.length; c++)
            qt(Ge[c], e);
        c = r;
        break;
    case "source":
        qt("error", e),
        c = r;
        break;
    case "img":
    case "image":
    case "link":
        qt("error", e),
        qt("load", e),
        c = r;
        break;
    case "form":
        qt("reset", e),
        qt("submit", e),
        c = r;
        break;
    case "details":
        qt("toggle", e),
        c = r;
        break;
    case "input":
        Te(e, r),
        c = xe(e, r),
        qt("invalid", e),
        cn(n, "onChange");
        break;
    case "option":
        c = Pe(e, r);
        break;
    case "select":
        e._wrapperState = {
            wasMultiple: !!r.multiple
        },
        c = l({}, r, {
```

```
                                value: void 0
                            }),
                            qt("invalid", e),
                            cn(n, "onChange");
                            break;
                        case "textarea":
                            Oe(e, r),
                            c = ze(e, r),
                            qt("invalid", e),
                            cn(n, "onChange");
                            break;
                        default:
                            c = r
                        }
                        an(i, c);
                        var s = c;
                        for (o in s)
                            if (s.hasOwnProperty(o)) {
                                var f = s[o];
                                "style" === o ? rn(e, f) : "dangerouslySetInnerHTML" === o ? null != (f = f ?
f.__html : void 0) && Ve(e, f) : "children" === o ? "string" === typeof f ? ("textarea" !== i || "" !== f) && We(e,
f) : "number" === typeof f && We(e, "" + f) : "suppressContentEditableWarning" !== o && "suppressHydrationWarning"
!== o && "autoFocus" !== o && (E.hasOwnProperty(o) ? null != f && cn(n, o) : null != f && G(e, o, f, u))
                            }
                        switch (i) {
                        case "input":
                            we(e),
                            Ce(e, r, !1);
                            break;
                        case "textarea":
                            we(e),
                            Ie(e);
                            break;
                        case "option":
                            null != r.value && e.setAttribute("value", "" + ge(r.value));
                            break;
                        case "select":
                            e.multiple = !!r.multiple,
                            null != (n = r.value) ? Ne(e, !!r.multiple, n, !1) : null != r.defaultValue && Ne(e,
!!r.multiple, r.defaultValue, !0);
                            break;
                        default:
                            "function" === typeof c.onClick && (e.onclick = sn)
                        }
                        kn(i, r) && (t.effectTag |= 4)
                    }
                    null !== t.ref && (t.effectTag |= 128)
                }
                return null;
            case 6:
                if (e && null != t.stateNode)
                    Ba(0, t, e.memoizedProps, r);
                else {
                    if ("string" !== typeof r && null === t.stateNode)
                        throw Error(a(166));
                    n = Mi(Ii.current),
                    Mi(Oi.current),
                    za(t) ? (n = t.stateNode,
                    r = t.memoizedProps,
                    n[Pn] = t,
                    n.nodeValue !== r && (t.effectTag |= 4)) : ((n = (9 === n.nodeType ? n :
n.ownerDocument).createTextNode(r))[Pn] = t,
                    t.stateNode = n)
                }
                return null;
            case 13:
                return dl(Ui),
                r = t.memoizedState,
                0 !== (64 & t.effectTag) ? (t.expirationTime = n,
```

```
                t) : (n = null !== r,
                r = !1,
                null === e ? void 0 !== t.memoizedProps.fallback && za(t) : (r = null !== (i = e.memoizedState),
                n || null === i || null !== (i = e.child.sibling) && (null !== (o = t.firstEffect) ? (t.firstEffect
= i,
                i.nextEffect = o) : (t.firstEffect = t.lastEffect = i,
                i.nextEffect = null),
                i.effectTag = 8)),
                n && !r && 0 !== (2 & t.mode) && (null === e && !0 !== t.memoizedProps.unstable_avoidThisFallback ||
0 !== (1 & Ui.current) ? Fo === Po && (Fo = No) : (Fo !== Po && Fo !== No || (Fo = zo),
                0 !== Vo && null !== Ro && (Wu(Ro, Mo),
                ju(Ro, Vo)))),
                (n || r) && (t.effectTag |= 4),
                null);
            case 4:
                return Di(),
                null;
            case 10:
                return li(t),
                null;
            case 17:
                return bl(t.type) && wl(),
                null;
            case 19:
                if (dl(Ui),
                null === (r = t.memoizedState))
                    return null;
                if (i = 0 !== (64 & t.effectTag),
                null === (o = r.rendering)) {
                    if (i)
                        Za(r, !1);
                    else if (Fo !== Po || null !== e && 0 !== (64 & e.effectTag))
                        for (o = t.child; null !== o; ) {
                            if (null !== (e = Vi(o))) {
                                for (t.effectTag |= 64,
                                Za(r, !1),
                                null !== (i = e.updateQueue) && (t.updateQueue = i,
                                t.effectTag |= 4),
                                null === r.lastEffect && (t.firstEffect = null),
                                t.lastEffect = r.lastEffect,
                                r = t.child; null !== r; )
                                    o = n,
                                    (i = r).effectTag &= 2,
                                    i.nextEffect = null,
                                    i.firstEffect = null,
                                    i.lastEffect = null,
                                    null === (e = i.alternate) ? (i.childExpirationTime = 0,
                                    i.expirationTime = o,
                                    i.child = null,
                                    i.memoizedProps = null,
                                    i.memoizedState = null,
                                    i.updateQueue = null,
                                    i.dependencies = null) : (i.childExpirationTime = e.childExpirationTime,
                                    i.expirationTime = e.expirationTime,
                                    i.child = e.child,
                                    i.memoizedProps = e.memoizedProps,
                                    i.memoizedState = e.memoizedState,
                                    i.updateQueue = e.updateQueue,
                                    o = e.dependencies,
                                    i.dependencies = null === o ? null : {
                                        expirationTime: o.expirationTime,
                                        firstContext: o.firstContext,
                                        responders: o.responders
                                    }),
                                    r = r.sibling;
                                return pl(Ui, 1 & Ui.current | 2),
                                t.child
                            }
                            o = o.sibling
```

```
                }
            } else {
                if (!i)
                    if (null !== (e = Vi(o))) {
                        if (t.effectTag |= 64,
                        i = !0,
                        null !== (n = e.updateQueue) && (t.updateQueue = n,
                        t.effectTag |= 4),
                        Za(r, !0),
                        null === r.tail && "hidden" === r.tailMode && !o.alternate)
                            return null !== (t = t.lastEffect = r.lastEffect) && (t.nextEffect = null),
                            null
                    } else
                        2 * Ql() - r.renderingStartTime > r.tailExpiration && 1 < n && (t.effectTag |= 64,
                        i = !0,
                        Za(r, !1),
                        t.expirationTime = t.childExpirationTime = n - 1);
                r.isBackwards ? (o.sibling = t.child,
                t.child = o) : (null !== (n = r.last) ? n.sibling = o : t.child = o,
                r.last = o)
            }
            return null !== r.tail ? (0 === r.tailExpiration && (r.tailExpiration = Ql() + 500),
            n = r.tail,
            r.rendering = n,
            r.tail = n.sibling,
            r.lastEffect = t.lastEffect,
            r.renderingStartTime = Ql(),
            n.sibling = null,
            t = Ui.current,
            pl(Ui, i ? 1 & t | 2 : 1 & t),
            n) : null
        }
        throw Error(a(156, t.tag))
    }
    function eo(e) {
        switch (e.tag) {
        case 1:
            bl(e.type) && wl();
            var t = e.effectTag;
            return 4096 & t ? (e.effectTag = -4097 & t | 64,
            e) : null;
        case 3:
            if (Di(),
            dl(vl),
            dl(hl),
            0 !== (64 & (t = e.effectTag)))
                throw Error(a(285));
            return e.effectTag = -4097 & t | 64,
            e;
        case 5:
            return Ai(e),
            null;
        case 13:
            return dl(Ui),
            4096 & (t = e.effectTag) ? (e.effectTag = -4097 & t | 64,
            e) : null;
        case 19:
            return dl(Ui),
            null;
        case 4:
            return Di(),
            null;
        case 10:
            return li(e),
            null;
        default:
            return null
        }
    }
```

```
function to(e, t) {
    return {
        value: e,
        source: t,
        stack: ye(t)
    }
}
Qa = function(e, t) {
    for (var n = t.child; null !== n; ) {
        if (5 === n.tag || 6 === n.tag)
            e.appendChild(n.stateNode);
        else if (4 !== n.tag && null !== n.child) {
            n.child.return = n,
            n = n.child;
            continue
        }
        if (n === t)
            break;
        for (; null === n.sibling; ) {
            if (null === n.return || n.return === t)
                return;
            n = n.return
        }
        n.sibling.return = n.return,
        n = n.sibling
    }
}
,
$a = function(e, t, n, r, i) {
    var a = e.memoizedProps;
    if (a !== r) {
        var o, u, c = t.stateNode;
        switch (Mi(Oi.current),
        e = null,
        n) {
        case "input":
            a = xe(c, a),
            r = xe(c, r),
            e = [];
            break;
        case "option":
            a = Pe(c, a),
            r = Pe(c, r),
            e = [];
            break;
        case "select":
            a = l({}, a, {
                value: void 0
            }),
            r = l({}, r, {
                value: void 0
            }),
            e = [];
            break;
        case "textarea":
            a = ze(c, a),
            r = ze(c, r),
            e = [];
            break;
        default:
            "function" !== typeof a.onClick && "function" === typeof r.onClick && (c.onclick = sn)
        }
        for (o in an(n, r),
        n = null,
        a)
            if (!r.hasOwnProperty(o) && a.hasOwnProperty(o) && null != a[o])
                if ("style" === o)
                    for (u in c = a[o])
                        c.hasOwnProperty(u) && (n || (n = {}),
```

```
                            n[u] = "");
                    else
                        "dangerouslySetInnerHTML" !== o && "children" !== o && "suppressContentEditableWarning"
!== o && "suppressHydrationWarning" !== o && "autoFocus" !== o && (E.hasOwnProperty(o) ? e || (e = []) : (e = e ||
[]).push(o, null));
                for (o in r) {
                    var s = r[o];
                    if (c = null != a ? a[o] : void 0,
                    r.hasOwnProperty(o) && s !== c && (null != s || null != c))
                        if ("style" === o)
                            if (c) {
                                for (u in c)
                                    !c.hasOwnProperty(u) || s && s.hasOwnProperty(u) || (n || (n = {}),
                                    n[u] = "");
                                for (u in s)
                                    s.hasOwnProperty(u) && c[u] !== s[u] && (n || (n = {}),
                                    n[u] = s[u])
                            } else
                                n || (e || (e = []),
                                e.push(o, n)),
                                n = s;
                        else
                            "dangerouslySetInnerHTML" === o ? (s = s ? s.__html : void 0,
                            c = c ? c.__html : void 0,
                            null != s && c !== s && (e = e || []).push(o, s)) : "children" === o ? c === s ||
"string" !== typeof s && "number" !== typeof s || (e = e || []).push(o, "" + s) : "suppressContentEditableWarning"
!== o && "suppressHydrationWarning" !== o && (E.hasOwnProperty(o) ? (null != s && cn(i, o),
                            e || c === s || (e = [])) : (e = e || []).push(o, s))
                }
                n && (e = e || []).push("style", n),
                i = e,
                (t.updateQueue = i) && (t.effectTag |= 4)
            }
        }
        ,
        Ba = function(e, t, n, r) {
            n !== r && (t.effectTag |= 4)
        }
        ;
        var no = "function" === typeof WeakSet ? WeakSet : Set;
        function ro(e, t) {
            var n = t.source
              , r = t.stack;
            null === r && null !== n && (r = ye(n)),
            null !== n && ve(n.type),
            t = t.value,
            null !== e && 1 === e.tag && ve(e.type);
            try {
                console.error(t)
            } catch (l) {
                setTimeout((function() {
                    throw l
                }
                ))
            }
        }
        function lo(e) {
            var t = e.ref;
            if (null !== t)
                if ("function" === typeof t)
                    try {
                        t(null)
                    } catch (n) {
                        Cu(e, n)
                    }
                else
                    t.current = null
        }
        function io(e, t) {
```

```
    switch (t.tag) {
    case 0:
    case 11:
    case 15:
    case 22:
        return;
    case 1:
        if (256 & t.effectTag && null !== e) {
            var n = e.memoizedProps
              , r = e.memoizedState;
            t = (e = t.stateNode).getSnapshotBeforeUpdate(t.elementType === t.type ? n : Zl(t.type, n), r),
            e.__reactInternalSnapshotBeforeUpdate = t
        }
        return;
    case 3:
    case 5:
    case 6:
    case 4:
    case 17:
        return
    }
    throw Error(a(163))
}
function ao(e, t) {
    if (null !== (t = null !== (t = t.updateQueue) ? t.lastEffect : null)) {
        var n = t = t.next;
        do {
            if ((n.tag & e) === e) {
                var r = n.destroy;
                n.destroy = void 0,
                void 0 !== r && r()
            }
            n = n.next
        } while (n !== t)
    }
}
function oo(e, t) {
    if (null !== (t = null !== (t = t.updateQueue) ? t.lastEffect : null)) {
        var n = t = t.next;
        do {
            if ((n.tag & e) === e) {
                var r = n.create;
                n.destroy = r()
            }
            n = n.next
        } while (n !== t)
    }
}
function uo(e, t, n) {
    switch (n.tag) {
    case 0:
    case 11:
    case 15:
    case 22:
        return void oo(3, n);
    case 1:
        if (e = n.stateNode,
        4 & n.effectTag)
            if (null === t)
                e.componentDidMount();
            else {
                var r = n.elementType === n.type ? t.memoizedProps : Zl(n.type, t.memoizedProps);
                e.componentDidUpdate(r, t.memoizedState, e.__reactInternalSnapshotBeforeUpdate)
            }
        return void (null !== (t = n.updateQueue) && hi(n, t, e));
    case 3:
        if (null !== (t = n.updateQueue)) {
            if (e = null,
            null !== n.child)
```

```
                switch (n.child.tag) {
                case 5:
                    e = n.child.stateNode;
                    break;
                case 1:
                    e = n.child.stateNode
                }
            hi(n, t, e)
        }
        return;
    case 5:
        return e = n.stateNode,
        void (null === t && 4 & n.effectTag && kn(n.type, n.memoizedProps) && e.focus());
    case 6:
    case 4:
    case 12:
        return;
    case 13:
        return void (null === n.memoizedState && (n = n.alternate,
        null !== n && (n = n.memoizedState,
        null !== n && (n = n.dehydrated,
        null !== n && Lt(n)))));
    case 19:
    case 17:
    case 20:
    case 21:
        return
    }
    throw Error(a(163))
}
function co(e, t, n) {
    switch ("function" === typeof zu && zu(t),
    t.tag) {
    case 0:
    case 11:
    case 14:
    case 15:
    case 22:
        if (null !== (e = t.updateQueue) && null !== (e = e.lastEffect)) {
            var r = e.next;
            Hl(97 < n ? 97 : n, (function() {
                var e = r;
                do {
                    var n = e.destroy;
                    if (void 0 !== n) {
                        var l = t;
                        try {
                            n()
                        } catch (i) {
                            Cu(l, i)
                        }
                    }
                    e = e.next
                } while (e !== r)
            }
            ))
        }
        break;
    case 1:
        lo(t),
        "function" === typeof (n = t.stateNode).componentWillUnmount && function(e, t) {
            try {
                t.props = e.memoizedProps,
                t.state = e.memoizedState,
                t.componentWillUnmount()
            } catch (n) {
                Cu(e, n)
            }
        }(t, n);
```

```
        break;
    case 5:
        lo(t);
        break;
    case 4:
        vo(e, t, n)
    }
}
function so(e) {
    var t = e.alternate;
    e.return = null,
    e.child = null,
    e.memoizedState = null,
    e.updateQueue = null,
    e.dependencies = null,
    e.alternate = null,
    e.firstEffect = null,
    e.lastEffect = null,
    e.pendingProps = null,
    e.memoizedProps = null,
    e.stateNode = null,
    null !== t && so(t)
}
function fo(e) {
    return 5 === e.tag || 3 === e.tag || 4 === e.tag
}
function po(e) {
    e: {
        for (var t = e.return; null !== t; ) {
            if (fo(t)) {
                var n = t;
                break e
            }
            t = t.return
        }
        throw Error(a(160))
    }
    switch (t = n.stateNode,
    n.tag) {
    case 5:
        var r = !1;
        break;
    case 3:
    case 4:
        t = t.containerInfo,
        r = !0;
        break;
    default:
        throw Error(a(161))
    }
    16 & n.effectTag && (We(t, ""),
    n.effectTag &= -17);
    e: t: for (n = e; ; ) {
        for (; null === n.sibling; ) {
            if (null === n.return || fo(n.return)) {
                n = null;
                break e
            }
            n = n.return
        }
        for (n.sibling.return = n.return,
        n = n.sibling; 5 !== n.tag && 6 !== n.tag && 18 !== n.tag; ) {
            if (2 & n.effectTag)
                continue t;
            if (null === n.child || 4 === n.tag)
                continue t;
            n.child.return = n,
            n = n.child
        }
```

```
                if (!(2 & n.effectTag)) {
                    n = n.stateNode;
                    break e
                }
            }
            r ? mo(e, n, t) : ho(e, n, t)
        }
        function mo(e, t, n) {
            var r = e.tag
              , l = 5 === r || 6 === r;
            if (l)
                e = l ? e.stateNode : e.stateNode.instance,
                t ? 8 === n.nodeType ? n.parentNode.insertBefore(e, t) : n.insertBefore(e, t) : (8 === n.nodeType ?
(t = n.parentNode).insertBefore(e, n) : (t = n).appendChild(e),
                null !== (n = n._reactRootContainer) && void 0 !== n || null !== t.onclick || (t.onclick = sn));
            else if (4 !== r && null !== (e = e.child))
                for (mo(e, t, n),
                e = e.sibling; null !== e; )
                    mo(e, t, n),
                    e = e.sibling
        }
        function ho(e, t, n) {
            var r = e.tag
              , l = 5 === r || 6 === r;
            if (l)
                e = l ? e.stateNode : e.stateNode.instance,
                t ? n.insertBefore(e, t) : n.appendChild(e);
            else if (4 !== r && null !== (e = e.child))
                for (ho(e, t, n),
                e = e.sibling; null !== e; )
                    ho(e, t, n),
                    e = e.sibling
        }
        function vo(e, t, n) {
            for (var r, l, i = t, o = !1; ; ) {
                if (!o) {
                    o = i.return;
                    e: for (; ; ) {
                        if (null === o)
                            throw Error(a(160));
                        switch (r = o.stateNode,
                        o.tag) {
                        case 5:
                            l = !1;
                            break e;
                        case 3:
                        case 4:
                            r = r.containerInfo,
                            l = !0;
                            break e
                        }
                        o = o.return
                    }
                    o = !0
                }
                if (5 === i.tag || 6 === i.tag) {
                    e: for (var u = e, c = i, s = n, f = c; ; )
                        if (co(u, f, s),
                        null !== f.child && 4 !== f.tag)
                            f.child.return = f,
                            f = f.child;
                        else {
                            if (f === c)
                                break e;
                            for (; null === f.sibling; ) {
                                if (null === f.return || f.return === c)
                                    break e;
                                f = f.return
                            }
```

```
                            f.sibling.return = f.return,
                            f = f.sibling
                        }
                    l ? (u = r,
                    c = i.stateNode,
                    8 === u.nodeType ? u.parentNode.removeChild(c) : u.removeChild(c)) : r.removeChild(i.stateNode)
                } else if (4 === i.tag) {
                    if (null !== i.child) {
                        r = i.stateNode.containerInfo,
                        l = !0,
                        i.child.return = i,
                        i = i.child;
                        continue
                    }
                } else if (co(e, i, n),
                null !== i.child) {
                    i.child.return = i,
                    i = i.child;
                    continue
                }
                if (i === t)
                    break;
                for (; null === i.sibling; ) {
                    if (null === i.return || i.return === t)
                        return;
                    4 === (i = i.return).tag && (o = !1)
                }
                i.sibling.return = i.return,
                i = i.sibling
            }
        }
        function yo(e, t) {
            switch (t.tag) {
            case 0:
            case 11:
            case 14:
            case 15:
            case 22:
                return void ao(3, t);
            case 1:
                return;
            case 5:
                var n = t.stateNode;
                if (null != n) {
                    var r = t.memoizedProps
                      , l = null !== e ? e.memoizedProps : r;
                    e = t.type;
                    var i = t.updateQueue;
                    if (t.updateQueue = null,
                    null !== i) {
                        for (n[Nn] = r,
                        "input" === e && "radio" === r.type && null != r.name && Ee(n, r),
                        on(e, l),
                        t = on(e, r),
                        l = 0; l < i.length; l += 2) {
                            var o = i[l]
                              , u = i[l + 1];
                            "style" === o ? rn(n, u) : "dangerouslySetInnerHTML" === o ? Ve(n, u) : "children" === o
? We(n, u) : G(n, o, u, t)
                        }
                        switch (e) {
                        case "input":
                            Se(n, r);
                            break;
                        case "textarea":
                            Re(n, r);
                            break;
                        case "select":
                            t = n._wrapperState.wasMultiple,
```

```
                            n._wrapperState.wasMultiple = !!r.multiple,
                            null != (e = r.value) ? Ne(n, !!r.multiple, e, !1) : t !== !!r.multiple && (null !=
r.defaultValue ? Ne(n, !!r.multiple, r.defaultValue, !0) : Ne(n, !!r.multiple, r.multiple ? [] : "", !1))
                        }
                    }
                }
                return;
            case 6:
                if (null === t.stateNode)
                    throw Error(a(162));
                return void (t.stateNode.nodeValue = t.memoizedProps);
            case 3:
                return void ((t = t.stateNode).hydrate && (t.hydrate = !1,
                Lt(t.containerInfo)));
            case 12:
                return;
            case 13:
                if (n = t,
                null === t.memoizedState ? r = !1 : (r = !0,
                n = t.child,
                jo = Ql()),
                null !== n)
                    e: for (e = n; ; ) {
                        if (5 === e.tag)
                            i = e.stateNode,
                            r ? "function" === typeof (i = i.style).setProperty ? i.setProperty("display", "none",
"important") : i.display = "none" : (i = e.stateNode,
                            l = void 0 !== (l = e.memoizedProps.style) && null !== l && l.hasOwnProperty("display")
? l.display : null,
                            i.style.display = nn("display", l));
                        else if (6 === e.tag)
                            e.stateNode.nodeValue = r ? "" : e.memoizedProps;
                        else {
                            if (13 === e.tag && null !== e.memoizedState && null === e.memoizedState.dehydrated) {
                                (i = e.child.sibling).return = e,
                                e = i;
                                continue
                            }
                            if (null !== e.child) {
                                e.child.return = e,
                                e = e.child;
                                continue
                            }
                        }
                        if (e === n)
                            break;
                        for (; null === e.sibling; ) {
                            if (null === e.return || e.return === n)
                                break e;
                            e = e.return
                        }
                        e.sibling.return = e.return,
                        e = e.sibling
                    }
                return void go(t);
            case 19:
                return void go(t);
            case 17:
                return
            }
            throw Error(a(163))
        }
        function go(e) {
            var t = e.updateQueue;
            if (null !== t) {
                e.updateQueue = null;
                var n = e.stateNode;
                null === n && (n = e.stateNode = new no),
                t.forEach((function(t) {
```

```
                var r = Pu.bind(null, e, t);
                n.has(t) || (n.add(t),
                t.then(r, r))
            }
            ))
        }
    }
    var bo = "function" === typeof WeakMap ? WeakMap : Map;
    function wo(e, t, n) {
        (n = fi(n, null)).tag = 3,
        n.payload = {
            element: null
        };
        var r = t.value;
        return n.callback = function() {
            $o || ($o = !0,
            Bo = r),
            ro(e, t)
        }
        ,
        n
    }
    function ko(e, t, n) {
        (n = fi(n, null)).tag = 3;
        var r = e.type.getDerivedStateFromError;
        if ("function" === typeof r) {
            var l = t.value;
            n.payload = function() {
                return ro(e, t),
                r(l)
            }
        }
        var i = e.stateNode;
        return null !== i && "function" === typeof i.componentDidCatch && (n.callback = function() {
            "function" !== typeof r && (null === Ho ? Ho = new Set([this]) : Ho.add(this),
            ro(e, t));
            var n = t.stack;
            this.componentDidCatch(t.value, {
                componentStack: null !== n ? n : ""
            })
        }
        ),
        n
    }
    var xo, To = Math.ceil, Eo = X.ReactCurrentDispatcher, So = X.ReactCurrentOwner, Co = 16, _o = 32, Po = 0,
No = 3, zo = 4, Oo = 0, Ro = null, Io = null, Mo = 0, Fo = Po, Do = null, Lo = 1073741823, Ao = 1073741823, Uo =
null, Vo = 0, Wo = !1, jo = 0, Qo = null, $o = !1, Bo = null, Ho = null, Ko = !1, qo = null, Yo = 90, Xo = null, Go
= 0, Zo = null, Jo = 0;
    function eu() {
        return 0 !== (48 & Oo) ? 1073741821 - (Ql() / 10 | 0) : 0 !== Jo ? Jo : Jo = 1073741821 - (Ql() / 10 |
0)
    }
    function tu(e, t, n) {
        if (0 === (2 & (t = t.mode)))
            return 1073741823;
        var r = $l();
        if (0 === (4 & t))
            return 99 === r ? 1073741823 : 1073741822;
        if (0 !== (Oo & Co))
            return Mo;
        if (null !== n)
            e = Gl(e, 0 | n.timeoutMs || 5e3, 250);
        else
            switch (r) {
            case 99:
                e = 1073741823;
                break;
            case 98:
                e = Gl(e, 150, 100);
```

```
                    break;
                case 97:
                case 96:
                    e = Gl(e, 5e3, 250);
                    break;
                case 95:
                    e = 2;
                    break;
                default:
                    throw Error(a(326))
                }
            return null !== Ro && e === Mo && --e,
            e
        }
        function nu(e, t) {
            if (50 < Go)
                throw Go = 0,
                Zo = null,
                Error(a(185));
            if (null !== (e = ru(e, t))) {
                var n = $l();
                1073741823 === t ? 0 !== (8 & Oo) && 0 === (48 & Oo) ? ou(e) : (iu(e),
                0 === Oo && Yl()) : iu(e),
                0 === (4 & Oo) || 98 !== n && 99 !== n || (null === Xo ? Xo = new Map([[e, t]]) : (void 0 === (n =
Xo.get(e)) || n > t) && Xo.set(e, t))
            }
        }
        function ru(e, t) {
            e.expirationTime < t && (e.expirationTime = t);
            var n = e.alternate;
            null !== n && n.expirationTime < t && (n.expirationTime = t);
            var r = e.return
              , l = null;
            if (null === r && 3 === e.tag)
                l = e.stateNode;
            else
                for (; null !== r; ) {
                    if (n = r.alternate,
                    r.childExpirationTime < t && (r.childExpirationTime = t),
                    null !== n && n.childExpirationTime < t && (n.childExpirationTime = t),
                    null === r.return && 3 === r.tag) {
                        l = r.stateNode;
                        break
                    }
                    r = r.return
                }
            return null !== l && (Ro === l && (mu(t),
            Fo === zo && Wu(l, Mo)),
            ju(l, t)),
            l
        }
        function lu(e) {
            var t = e.lastExpiredTime;
            if (0 !== t)
                return t;
            if (!Vu(e, t = e.firstPendingTime))
                return t;
            var n = e.lastPingedTime;
            return 2 >= (e = n > (e = e.nextKnownPendingLevel) ? n : e) && t !== e ? 0 : e
        }
        function iu(e) {
            if (0 !== e.lastExpiredTime)
                e.callbackExpirationTime = 1073741823,
                e.callbackPriority = 99,
                e.callbackNode = ql(ou.bind(null, e));
            else {
                var t = lu(e)
                  , n = e.callbackNode;
                if (0 === t)
```

```
                    null !== n && (e.callbackNode = null,
                    e.callbackExpirationTime = 0,
                    e.callbackPriority = 90);
                else {
                    var r = eu();
                    if (1073741823 === t ? r = 99 : 1 === t || 2 === t ? r = 95 : r = 0 >= (r = 10 * (1073741821 -
t) - 10 * (1073741821 - r)) ? 99 : 250 >= r ? 98 : 5250 >= r ? 97 : 95,
                    null !== n) {
                        var l = e.callbackPriority;
                        if (e.callbackExpirationTime === t && l >= r)
                            return;
                        n !== Dl && _l(n)
                    }
                    e.callbackExpirationTime = t,
                    e.callbackPriority = r,
                    t = 1073741823 === t ? ql(ou.bind(null, e)) : Kl(r, au.bind(null, e), {
                        timeout: 10 * (1073741821 - t) - Ql()
                    }),
                    e.callbackNode = t
                }
            }
        }
        function au(e, t) {
            if (Jo = 0,
            t)
                return Qu(e, t = eu()),
                iu(e),
                null;
            var n = lu(e);
            if (0 !== n) {
                if (t = e.callbackNode,
                0 !== (48 & Oo))
                    throw Error(a(327));
                if (Tu(),
                e === Ro && n === Mo || su(e, n),
                null !== Io) {
                    var r = Oo;
                    Oo |= Co;
                    for (var l = du(); ; )
                        try {
                            vu();
                            break
                        } catch (u) {
                            fu(e, u)
                        }
                    if (ri(),
                    Oo = r,
                    Eo.current = l,
                    1 === Fo)
                        throw t = Do,
                        su(e, n),
                        Wu(e, n),
                        iu(e),
                        t;
                    if (null === Io)
                        switch (l = e.finishedWork = e.current.alternate,
                        e.finishedExpirationTime = n,
                        r = Fo,
                        Ro = null,
                        r) {
                        case Po:
                        case 1:
                            throw Error(a(345));
                        case 2:
                            Qu(e, 2 < n ? 2 : n);
                            break;
                        case No:
                            if (Wu(e, n),
                            n === (r = e.lastSuspendedTime) && (e.nextKnownPendingLevel = bu(l)),
```

```
                            1073741823 === Lo && 10 < (l = jo + 500 - Ql())) {
                                if (Wo) {
                                    var i = e.lastPingedTime;
                                    if (0 === i || i >= n) {
                                        e.lastPingedTime = n,
                                        su(e, n);
                                        break
                                    }
                                }
                                if (0 !== (i = lu(e)) && i !== n)
                                    break;
                                if (0 !== r && r !== n) {
                                    e.lastPingedTime = r;
                                    break
                                }
                                e.timeoutHandle = Tn(wu.bind(null, e), l);
                                break
                            }
                            wu(e);
                            break;
                        case zo:
                            if (Wu(e, n),
                            n === (r = e.lastSuspendedTime) && (e.nextKnownPendingLevel = bu(l)),
                            Wo && (0 === (l = e.lastPingedTime) || l >= n)) {
                                e.lastPingedTime = n,
                                su(e, n);
                                break
                            }
                            if (0 !== (l = lu(e)) && l !== n)
                                break;
                            if (0 !== r && r !== n) {
                                e.lastPingedTime = r;
                                break
                            }
                            if (1073741823 !== Ao ? r = 10 * (1073741821 - Ao) - Ql() : 1073741823 === Lo ? r = 0 :
(r = 10 * (1073741821 - Lo) - 5e3,
                            0 > (r = (l = Ql()) - r) && (r = 0),
                            (n = 10 * (1073741821 - n) - l) < (r = (120 > r ? 120 : 480 > r ? 480 : 1080 > r ? 1080
: 1920 > r ? 1920 : 3e3 > r ? 3e3 : 4320 > r ? 4320 : 1960 * To(r / 1960)) - r) && (r = n)),
                            10 < r) {
                                e.timeoutHandle = Tn(wu.bind(null, e), r);
                                break
                            }
                            wu(e);
                            break;
                        case 5:
                            if (1073741823 !== Lo && null !== Uo) {
                                i = Lo;
                                var o = Uo;
                                if (0 >= (r = 0 | o.busyMinDurationMs) ? r = 0 : (l = 0 | o.busyDelayMs,
                                r = (i = Ql() - (10 * (1073741821 - i) - (0 | o.timeoutMs || 5e3))) <= l ? 0 : l + r
- i),
                                10 < r) {
                                    Wu(e, n),
                                    e.timeoutHandle = Tn(wu.bind(null, e), r);
                                    break
                                }
                            }
                            wu(e);
                            break;
                        default:
                            throw Error(a(329))
                        }
                    if (iu(e),
                    e.callbackNode === t)
                        return au.bind(null, e)
                }
            }
            return null
```

```
}
function ou(e) {
    var t = e.lastExpiredTime;
    if (t = 0 !== t ? t : 1073741823,
    0 !== (48 & Oo))
        throw Error(a(327));
    if (Tu(),
    e === Ro && t === Mo || su(e, t),
    null !== Io) {
        var n = Oo;
        Oo |= Co;
        for (var r = du(); ; )
            try {
                hu();
                break
            } catch (l) {
                fu(e, l)
            }
        if (ri(),
        Oo = n,
        Eo.current = r,
        1 === Fo)
            throw n = Do,
            su(e, t),
            Wu(e, t),
            iu(e),
            n;
        if (null !== Io)
            throw Error(a(261));
        e.finishedWork = e.current.alternate,
        e.finishedExpirationTime = t,
        Ro = null,
        wu(e),
        iu(e)
    }
    return null
}
function uu(e, t) {
    var n = Oo;
    Oo |= 1;
    try {
        return e(t)
    } finally {
        0 === (Oo = n) && Yl()
    }
}
function cu(e, t) {
    var n = Oo;
    Oo &= -2,
    Oo |= 8;
    try {
        return e(t)
    } finally {
        0 === (Oo = n) && Yl()
    }
}
function su(e, t) {
    e.finishedWork = null,
    e.finishedExpirationTime = 0;
    var n = e.timeoutHandle;
    if (-1 !== n && (e.timeoutHandle = -1,
    En(n)),
    null !== Io)
        for (n = Io.return; null !== n; ) {
            var r = n;
            switch (r.tag) {
            case 1:
                null !== (r = r.type.childContextTypes) && void 0 !== r && wl();
                break;
```

```
            case 3:
                Di(),
                dl(vl),
                dl(hl);
                break;
            case 5:
                Ai(r);
                break;
            case 4:
                Di();
                break;
            case 13:
            case 19:
                dl(Ui);
                break;
            case 10:
                li(r)
            }
            n = n.return
        }
    Ro = e,
    Io = Mu(e.current, null),
    Mo = t,
    Fo = Po,
    Do = null,
    Ao = Lo = 1073741823,
    Uo = null,
    Vo = 0,
    Wo = !1
}
function fu(e, t) {
    for (; ; ) {
        try {
            if (ri(),
            ji.current = ba,
            qi)
                for (var n = Bi.memoizedState; null !== n; ) {
                    var r = n.queue;
                    null !== r && (r.pending = null),
                    n = n.next
                }
            if ($i = 0,
            Ki = Hi = Bi = null,
            qi = !1,
            null === Io || null === Io.return)
                return Fo = 1,
                Do = t,
                Io = null;
            e: {
                var l = e
                  , i = Io.return
                  , a = Io
                  , o = t;
                if (t = Mo,
                a.effectTag |= 2048,
                a.firstEffect = a.lastEffect = null,
                null !== o && "object" === typeof o && "function" === typeof o.then) {
                    var u = o;
                    if (0 === (2 & a.mode)) {
                        var c = a.alternate;
                        c ? (a.updateQueue = c.updateQueue,
                        a.memoizedState = c.memoizedState,
                        a.expirationTime = c.expirationTime) : (a.updateQueue = null,
                        a.memoizedState = null)
                    }
                    var s = 0 !== (1 & Ui.current)
                      , f = i;
                    do {
                        var d;
```

```
                                if (d = 13 === f.tag) {
                                    var p = f.memoizedState;
                                    if (null !== p)
                                        d = null !== p.dehydrated;
                                    else {
                                        var m = f.memoizedProps;
                                        d = void 0 !== m.fallback && (!0 !== m.unstable_avoidThisFallback || !s)
                                    }
                                }
                                if (d) {
                                    var h = f.updateQueue;
                                    if (null === h) {
                                        var v = new Set;
                                        v.add(u),
                                        f.updateQueue = v
                                    } else
                                        h.add(u);
                                    if (0 === (2 & f.mode)) {
                                        if (f.effectTag |= 64,
                                        a.effectTag &= -2981,
                                        1 === a.tag)
                                            if (null === a.alternate)
                                                a.tag = 17;
                                            else {
                                                var y = fi(1073741823, null);
                                                y.tag = 2,
                                                di(a, y)
                                            }
                                        a.expirationTime = 1073741823;
                                        break e
                                    }
                                    o = void 0,
                                    a = t;
                                    var g = l.pingCache;
                                    if (null === g ? (g = l.pingCache = new bo,
                                    o = new Set,
                                    g.set(u, o)) : void 0 === (o = g.get(u)) && (o = new Set,
                                    g.set(u, o)),
                                    !o.has(a)) {
                                        o.add(a);
                                        var b = _u.bind(null, l, u, a);
                                        u.then(b, b)
                                    }
                                    f.effectTag |= 4096,
                                    f.expirationTime = t;
                                    break e
                                }
                                f = f.return
                            } while (null !== f);
                            o = Error((ve(a.type) || "A React component") + " suspended while rendering, but no
fallback UI was specified.\n\nAdd a <Suspense fallback=...> component higher in the tree to provide a loading
indicator or placeholder to display." + ye(a))
                        }
                        5 !== Fo && (Fo = 2),
                        o = to(o, a),
                        f = i;
                        do {
                            switch (f.tag) {
                            case 3:
                                u = o,
                                f.effectTag |= 4096,
                                f.expirationTime = t,
                                pi(f, wo(f, u, t));
                                break e;
                            case 1:
                                u = o;
                                var w = f.type
                                  , k = f.stateNode;
                                if (0 === (64 & f.effectTag) && ("function" === typeof w.getDerivedStateFromError ||
```

```
null !== k && "function" === typeof k.componentDidCatch && (null === Ho || !Ho.has(k))) {
                                    f.effectTag |= 4096,
                                    f.expirationTime = t,
                                    pi(f, ko(f, u, t));
                                    break e
                                }
                            }
                            f = f.return
                        } while (null !== f)
                    }
                    Io = gu(Io)
                } catch (x) {
                    t = x;
                    continue
                }
                break
            }
        }
        function du() {
            var e = Eo.current;
            return Eo.current = ba,
            null === e ? ba : e
        }
        function pu(e, t) {
            e < Lo && 2 < e && (Lo = e),
            null !== t && e < Ao && 2 < e && (Ao = e,
            Uo = t)
        }
        function mu(e) {
            e > Vo && (Vo = e)
        }
        function hu() {
            for (; null !== Io; )
                Io = yu(Io)
        }
        function vu() {
            for (; null !== Io && !Ll(); )
                Io = yu(Io)
        }
        function yu(e) {
            var t = xo(e.alternate, e, Mo);
            return e.memoizedProps = e.pendingProps,
            null === t && (t = gu(e)),
            So.current = null,
            t
        }
        function gu(e) {
            Io = e;
            do {
                var t = Io.alternate;
                if (e = Io.return,
                0 === (2048 & Io.effectTag)) {
                    if (t = Ja(t, Io, Mo),
                    1 === Mo || 1 !== Io.childExpirationTime) {
                        for (var n = 0, r = Io.child; null !== r; ) {
                            var l = r.expirationTime
                              , i = r.childExpirationTime;
                            l > n && (n = l),
                            i > n && (n = i),
                            r = r.sibling
                        }
                        Io.childExpirationTime = n
                    }
                    if (null !== t)
                        return t;
                    null !== e && 0 === (2048 & e.effectTag) && (null === e.firstEffect && (e.firstEffect =
Io.firstEffect),
                    null !== Io.lastEffect && (null !== e.lastEffect && (e.lastEffect.nextEffect = Io.firstEffect),
                    e.lastEffect = Io.lastEffect),
```

```
                    1 < Io.effectTag && (null !== e.lastEffect ? e.lastEffect.nextEffect = Io : e.firstEffect = Io,
                    e.lastEffect = Io))
                } else {
                    if (null !== (t = eo(Io)))
                        return t.effectTag &= 2047,
                        t;
                    null !== e && (e.firstEffect = e.lastEffect = null,
                    e.effectTag |= 2048)
                }
                if (null !== (t = Io.sibling))
                    return t;
                Io = e
            } while (null !== Io);
            return Fo === Po && (Fo = 5),
            null
        }
        function bu(e) {
            var t = e.expirationTime;
            return t > (e = e.childExpirationTime) ? t : e
        }
        function wu(e) {
            var t = $l();
            return Hl(99, ku.bind(null, e, t)),
            null
        }
        function ku(e, t) {
            do {
                Tu()
            } while (null !== qo);
            if (0 !== (48 & Oo))
                throw Error(a(327));
            var n = e.finishedWork
              , r = e.finishedExpirationTime;
            if (null === n)
                return null;
            if (e.finishedWork = null,
            e.finishedExpirationTime = 0,
            n === e.current)
                throw Error(a(177));
            e.callbackNode = null,
            e.callbackExpirationTime = 0,
            e.callbackPriority = 90,
            e.nextKnownPendingLevel = 0;
            var l = bu(n);
            if (e.firstPendingTime = l,
            r <= e.lastSuspendedTime ? e.firstSuspendedTime = e.lastSuspendedTime = e.nextKnownPendingLevel = 0 : r
<= e.firstSuspendedTime && (e.firstSuspendedTime = r - 1),
            r <= e.lastPingedTime && (e.lastPingedTime = 0),
            r <= e.lastExpiredTime && (e.lastExpiredTime = 0),
            e === Ro && (Io = Ro = null,
            Mo = 0),
            1 < n.effectTag ? null !== n.lastEffect ? (n.lastEffect.nextEffect = n,
            l = n.firstEffect) : l = n : l = n.firstEffect,
            null !== l) {
                var i = Oo;
                Oo |= _o,
                So.current = null,
                bn = Kt;
                var o = hn();
                if (vn(o)) {
                    if ("selectionStart"in o)
                        var u = {
                            start: o.selectionStart,
                            end: o.selectionEnd
                        };
                    else
                        e: {
                            var c = (u = (u = o.ownerDocument) && u.defaultView || window).getSelection &&
u.getSelection();
```

```
            if (c && 0 !== c.rangeCount) {
                u = c.anchorNode;
                var s = c.anchorOffset
                  , f = c.focusNode;
                c = c.focusOffset;
                try {
                    u.nodeType,
                    f.nodeType
                } catch (C) {
                    u = null;
                    break e
                }
                var d = 0
                  , p = -1
                  , m = -1
                  , h = 0
                  , v = 0
                  , y = o
                  , g = null;
                t: for (; ; ) {
                    for (var b; y !== u || 0 !== s && 3 !== y.nodeType || (p = d + s),
                    y !== f || 0 !== c && 3 !== y.nodeType || (m = d + c),
                    3 === y.nodeType && (d += y.nodeValue.length),
                    null !== (b = y.firstChild); )
                        g = y,
                        y = b;
                    for (; ; ) {
                        if (y === o)
                            break t;
                        if (g === u && ++h === s && (p = d),
                        g === f && ++v === c && (m = d),
                        null !== (b = y.nextSibling))
                            break;
                        g = (y = g).parentNode
                    }
                    y = b
                }
                u = -1 === p || -1 === m ? null : {
                    start: p,
                    end: m
                }
            } else
                u = null
        }
    u = u || {
        start: 0,
        end: 0
    }
} else
    u = null;
wn = {
    activeElementDetached: null,
    focusedElem: o,
    selectionRange: u
},
Kt = !1,
Qo = l;
do {
    try {
        xu()
    } catch (C) {
        if (null === Qo)
            throw Error(a(330));
        Cu(Qo, C),
        Qo = Qo.nextEffect
    }
} while (null !== Qo);
Qo = l;
do {
```

```
                    try {
                        for (o = e,
                        u = t; null !== Qo; ) {
                            var w = Qo.effectTag;
                            if (16 & w && We(Qo.stateNode, ""),
                            128 & w) {
                                var k = Qo.alternate;
                                if (null !== k) {
                                    var x = k.ref;
                                    null !== x && ("function" === typeof x ? x(null) : x.current = null)
                                }
                            }
                            switch (1038 & w) {
                            case 2:
                                po(Qo),
                                Qo.effectTag &= -3;
                                break;
                            case 6:
                                po(Qo),
                                Qo.effectTag &= -3,
                                yo(Qo.alternate, Qo);
                                break;
                            case 1024:
                                Qo.effectTag &= -1025;
                                break;
                            case 1028:
                                Qo.effectTag &= -1025,
                                yo(Qo.alternate, Qo);
                                break;
                            case 4:
                                yo(Qo.alternate, Qo);
                                break;
                            case 8:
                                vo(o, s = Qo, u),
                                so(s)
                            }
                            Qo = Qo.nextEffect
                        }
                    } catch (C) {
                        if (null === Qo)
                            throw Error(a(330));
                        Cu(Qo, C),
                        Qo = Qo.nextEffect
                    }
                } while (null !== Qo);
                if (x = wn,
                k = hn(),
                w = x.focusedElem,
                u = x.selectionRange,
                k !== w && w && w.ownerDocument && mn(w.ownerDocument.documentElement, w)) {
                    null !== u && vn(w) && (k = u.start,
                    void 0 === (x = u.end) && (x = k),
                    "selectionStart"in w ? (w.selectionStart = k,
                    w.selectionEnd = Math.min(x, w.value.length)) : (x = (k = w.ownerDocument || document) &&
k.defaultView || window).getSelection && (x = x.getSelection(),
                    s = w.textContent.length,
                    o = Math.min(u.start, s),
                    u = void 0 === u.end ? o : Math.min(u.end, s),
                    !x.extend && o > u && (s = u,
                    u = o,
                    o = s),
                    s = pn(w, o),
                    f = pn(w, u),
                    s && f && (1 !== x.rangeCount || x.anchorNode !== s.node || x.anchorOffset !== s.offset ||
x.focusNode !== f.node || x.focusOffset !== f.offset) && ((k = k.createRange()).setStart(s.node, s.offset),
                    x.removeAllRanges(),
                    o > u ? (x.addRange(k),
                    x.extend(f.node, f.offset)) : (k.setEnd(f.node, f.offset),
                    x.addRange(k))))),
```

```
        k = [];
        for (x = w; x = x.parentNode; )
            1 === x.nodeType && k.push({
                element: x,
                left: x.scrollLeft,
                top: x.scrollTop
            });
        for ("function" === typeof w.focus && w.focus(),
        w = 0; w < k.length; w++)
            (x = k[w]).element.scrollLeft = x.left,
            x.element.scrollTop = x.top
    }
    Kt = !!bn,
    wn = bn = null,
    e.current = n,
    Qo = l;
    do {
        try {
            for (w = e; null !== Qo; ) {
                var T = Qo.effectTag;
                if (36 & T && uo(w, Qo.alternate, Qo),
                128 & T) {
                    k = void 0;
                    var E = Qo.ref;
                    if (null !== E) {
                        var S = Qo.stateNode;
                        switch (Qo.tag) {
                        case 5:
                            k = S;
                            break;
                        default:
                            k = S
                        }
                        "function" === typeof E ? E(k) : E.current = k
                    }
                }
                Qo = Qo.nextEffect
            }
        } catch (C) {
            if (null === Qo)
                throw Error(a(330));
            Cu(Qo, C),
            Qo = Qo.nextEffect
        }
    } while (null !== Qo);
    Qo = null,
    Al(),
    Oo = i
} else
    e.current = n;
if (Ko)
    Ko = !1,
    qo = e,
    Yo = t;
else
    for (Qo = l; null !== Qo; )
        t = Qo.nextEffect,
        Qo.nextEffect = null,
        Qo = t;
if (0 === (t = e.firstPendingTime) && (Ho = null),
1073741823 === t ? e === Zo ? Go++ : (Go = 0,
Zo = e) : Go = 0,
"function" === typeof Nu && Nu(n.stateNode, r),
iu(e),
$o)
    throw $o = !1,
    e = Bo,
    Bo = null,
    e;
```

```
    return 0 !== (8 & Oo) || Yl(),
    null
}
function xu() {
    for (; null !== Qo; ) {
        var e = Qo.effectTag;
        0 !== (256 & e) && io(Qo.alternate, Qo),
        0 === (512 & e) || Ko || (Ko = !0,
        Kl(97, (function() {
            return Tu(),
            null
        }
        ))),
        Qo = Qo.nextEffect
    }
}
function Tu() {
    if (90 !== Yo) {
        var e = 97 < Yo ? 97 : Yo;
        return Yo = 90,
        Hl(e, Eu)
    }
}
function Eu() {
    if (null === qo)
        return !1;
    var e = qo;
    if (qo = null,
    0 !== (48 & Oo))
        throw Error(a(331));
    var t = Oo;
    for (Oo |= _o,
    e = e.current.firstEffect; null !== e; ) {
        try {
            var n = e;
            if (0 !== (512 & n.effectTag))
                switch (n.tag) {
                case 0:
                case 11:
                case 15:
                case 22:
                    ao(5, n),
                    oo(5, n)
                }
        } catch (r) {
            if (null === e)
                throw Error(a(330));
            Cu(e, r)
        }
        n = e.nextEffect,
        e.nextEffect = null,
        e = n
    }
    return Oo = t,
    Yl(),
    !0
}
function Su(e, t, n) {
    di(e, t = wo(e, t = to(n, t), 1073741823)),
    null !== (e = ru(e, 1073741823)) && iu(e)
}
function Cu(e, t) {
    if (3 === e.tag)
        Su(e, e, t);
    else
        for (var n = e.return; null !== n; ) {
            if (3 === n.tag) {
                Su(n, e, t);
                break
```

```
                    }
                    if (1 === n.tag) {
                        var r = n.stateNode;
                        if ("function" === typeof n.type.getDerivedStateFromError || "function" === typeof
r.componentDidCatch && (null === Ho || !Ho.has(r))) {
                            di(n, e = ko(n, e = to(t, e), 1073741823)),
                            null !== (n = ru(n, 1073741823)) && iu(n);
                            break
                        }
                    }
                    n = n.return
                }
        }
        function _u(e, t, n) {
            var r = e.pingCache;
            null !== r && r.delete(t),
            Ro === e && Mo === n ? Fo === zo || Fo === No && 1073741823 === Lo && Ql() - jo < 500 ? su(e, Mo) : Wo =
!0 : Vu(e, n) && (0 !== (t = e.lastPingedTime) && t < n || (e.lastPingedTime = n,
            iu(e)))
        }
        function Pu(e, t) {
            var n = e.stateNode;
            null !== n && n.delete(t),
            0 === (t = 0) && (t = tu(t = eu(), e, null)),
            null !== (e = ru(e, t)) && iu(e)
        }
        xo = function(e, t, n) {
            var r = t.expirationTime;
            if (null !== e) {
                var l = t.pendingProps;
                if (e.memoizedProps !== l || vl.current)
                    Ia = !0;
                else {
                    if (r < n) {
                        switch (Ia = !1,
                        t.tag) {
                        case 3:
                            ja(t),
                            Oa();
                            break;
                        case 5:
                            if (Li(t),
                            4 & t.mode && 1 !== n && l.hidden)
                                return t.expirationTime = t.childExpirationTime = 1,
                                null;
                            break;
                        case 1:
                            bl(t.type) && Tl(t);
                            break;
                        case 4:
                            Fi(t, t.stateNode.containerInfo);
                            break;
                        case 10:
                            r = t.memoizedProps.value,
                            l = t.type._context,
                            pl(Jl, l._currentValue),
                            l._currentValue = r;
                            break;
                        case 13:
                            if (null !== t.memoizedState)
                                return 0 !== (r = t.child.childExpirationTime) && r >= n ? Ka(e, t, n) : (pl(Ui, 1 &
Ui.current),
                                null !== (t = Ga(e, t, n)) ? t.sibling : null);
                            pl(Ui, 1 & Ui.current);
                            break;
                        case 19:
                            if (r = t.childExpirationTime >= n,
                            0 !== (64 & e.effectTag)) {
                                if (r)
```

```
                        return Xa(e, t, n);
                    t.effectTag |= 64
                }
                if (null !== (l = t.memoizedState) && (l.rendering = null,
                l.tail = null),
                pl(Ui, Ui.current),
                !r)
                    return null
            }
            return Ga(e, t, n)
        }
        Ia = !1
    }
} else
    Ia = !1;
switch (t.expirationTime = 0,
t.tag) {
case 2:
    if (r = t.type,
    null !== e && (e.alternate = null,
    t.alternate = null,
    t.effectTag |= 2),
    e = t.pendingProps,
    l = gl(t, hl.current),
    ai(t, n),
    l = Gi(null, t, r, e, l, n),
    t.effectTag |= 1,
    "object" === typeof l && null !== l && "function" === typeof l.render && void 0 === l.$$typeof) {
        if (t.tag = 1,
        t.memoizedState = null,
        t.updateQueue = null,
        bl(r)) {
            var i = !0;
            Tl(t)
        } else
            i = !1;
        t.memoizedState = null !== l.state && void 0 !== l.state ? l.state : null,
        ci(t);
        var o = r.getDerivedStateFromProps;
        "function" === typeof o && gi(t, r, o, e),
        l.updater = bi,
        t.stateNode = l,
        l._reactInternalFiber = t,
        Ti(t, r, e, n),
        t = Wa(null, t, r, !0, i, n)
    } else
        t.tag = 0,
        Ma(null, t, l, n),
        t = t.child;
    return t;
case 16:
    e: {
        if (l = t.elementType,
        null !== e && (e.alternate = null,
        t.alternate = null,
        t.effectTag |= 2),
        e = t.pendingProps,
        function(e) {
            if (-1 === e._status) {
                e._status = 0;
                var t = e._ctor;
                t = t(),
                e._result = t,
                t.then((function(t) {
                    0 === e._status && (t = t.default,
                    e._status = 1,
                    e._result = t)
                }
                ), (function(t) {
```

```
                    0 === e._status && (e._status = 2,
                    e._result = t)
                }
                ))
            }
        }(l),
        1 !== l._status)
            throw l._result;
        switch (l = l._result,
        t.type = l,
        i = t.tag = function(e) {
            if ("function" === typeof e)
                return Iu(e) ? 1 : 0;
            if (void 0 !== e && null !== e) {
                if ((e = e.$$typeof) === ue)
                    return 11;
                if (e === fe)
                    return 14
            }
            return 2
        }(l),
        e = Zl(l, e),
        i) {
        case 0:
            t = Ua(null, t, l, e, n);
            break e;
        case 1:
            t = Va(null, t, l, e, n);
            break e;
        case 11:
            t = Fa(null, t, l, e, n);
            break e;
        case 14:
            t = Da(null, t, l, Zl(l.type, e), r, n);
            break e
        }
        throw Error(a(306, l, ""))
    }
    return t;
case 0:
    return r = t.type,
    l = t.pendingProps,
    Ua(e, t, r, l = t.elementType === r ? l : Zl(r, l), n);
case 1:
    return r = t.type,
    l = t.pendingProps,
    Va(e, t, r, l = t.elementType === r ? l : Zl(r, l), n);
case 3:
    if (ja(t),
    r = t.updateQueue,
    null === e || null === r)
        throw Error(a(282));
    if (r = t.pendingProps,
    l = null !== (l = t.memoizedState) ? l.element : null,
    si(e, t),
    mi(t, r, null, n),
    (r = t.memoizedState.element) === l)
        Oa(),
        t = Ga(e, t, n);
    else {
        if ((l = t.stateNode.hydrate) && (Ea = Sn(t.stateNode.containerInfo.firstChild),
        Ta = t,
        l = Sa = !0),
        l)
            for (n = Ni(t, null, r, n),
            t.child = n; n; )
                n.effectTag = -3 & n.effectTag | 1024,
                n = n.sibling;
        else
```

```
                    Ma(e, t, r, n),
                    Oa();
                t = t.child
            }
            return t;
        case 5:
            return Li(t),
            null === e && Pa(t),
            r = t.type,
            l = t.pendingProps,
            i = null !== e ? e.memoizedProps : null,
            o = l.children,
            xn(r, l) ? o = null : null !== i && xn(r, i) && (t.effectTag |= 16),
            Aa(e, t),
            4 & t.mode && 1 !== n && l.hidden ? (t.expirationTime = t.childExpirationTime = 1,
            t = null) : (Ma(e, t, o, n),
            t = t.child),
            t;
        case 6:
            return null === e && Pa(t),
            null;
        case 13:
            return Ka(e, t, n);
        case 4:
            return Fi(t, t.stateNode.containerInfo),
            r = t.pendingProps,
            null === e ? t.child = Pi(t, null, r, n) : Ma(e, t, r, n),
            t.child;
        case 11:
            return r = t.type,
            l = t.pendingProps,
            Fa(e, t, r, l = t.elementType === r ? l : Zl(r, l), n);
        case 7:
            return Ma(e, t, t.pendingProps, n),
            t.child;
        case 8:
        case 12:
            return Ma(e, t, t.pendingProps.children, n),
            t.child;
        case 10:
            e: {
                r = t.type._context,
                l = t.pendingProps,
                o = t.memoizedProps,
                i = l.value;
                var u = t.type._context;
                if (pl(Jl, u._currentValue),
                u._currentValue = i,
                null !== o)
                    if (u = o.value,
                    0 === (i = Wr(u, i) ? 0 : 0 | ("function" === typeof r._calculateChangedBits ?
r._calculateChangedBits(u, i) : 1073741823))) {
                        if (o.children === l.children && !vl.current) {
                            t = Ga(e, t, n);
                            break e
                        }
                    } else
                        for (null !== (u = t.child) && (u.return = t); null !== u; ) {
                            var c = u.dependencies;
                            if (null !== c) {
                                o = u.child;
                                for (var s = c.firstContext; null !== s; ) {
                                    if (s.context === r && 0 !== (s.observedBits & i)) {
                                        1 === u.tag && ((s = fi(n, null)).tag = 2,
                                        di(u, s)),
                                        u.expirationTime < n && (u.expirationTime = n),
                                        null !== (s = u.alternate) && s.expirationTime < n && (s.expirationTime
= n),
                                        ii(u.return, n),
```

```
                                    c.expirationTime < n && (c.expirationTime = n);
                                    break
                                }
                                s = s.next
                            }
                        } else
                            o = 10 === u.tag && u.type === t.type ? null : u.child;
                        if (null !== o)
                            o.return = u;
                        else
                            for (o = u; null !== o; ) {
                                if (o === t) {
                                    o = null;
                                    break
                                }
                                if (null !== (u = o.sibling)) {
                                    u.return = o.return,
                                    o = u;
                                    break
                                }
                                o = o.return
                            }
                        u = o
                    }
                Ma(e, t, l.children, n),
                t = t.child
            }
            return t;
        case 9:
            return l = t.type,
            r = (i = t.pendingProps).children,
            ai(t, n),
            r = r(l = oi(l, i.unstable_observedBits)),
            t.effectTag |= 1,
            Ma(e, t, r, n),
            t.child;
        case 14:
            return i = Zl(l = t.type, t.pendingProps),
            Da(e, t, l, i = Zl(l.type, i), r, n);
        case 15:
            return La(e, t, t.type, t.pendingProps, r, n);
        case 17:
            return r = t.type,
            l = t.pendingProps,
            l = t.elementType === r ? l : Zl(r, l),
            null !== e && (e.alternate = null,
            t.alternate = null,
            t.effectTag |= 2),
            t.tag = 1,
            bl(r) ? (e = !0,
            Tl(t)) : e = !1,
            ai(t, n),
            ki(t, r, l),
            Ti(t, r, l, n),
            Wa(null, t, r, !0, e, n);
        case 19:
            return Xa(e, t, n)
        }
        throw Error(a(156, t.tag))
    }
    ;
    var Nu = null
      , zu = null;
    function Ou(e, t, n, r) {
        this.tag = e,
        this.key = n,
        this.sibling = this.child = this.return = this.stateNode = this.type = this.elementType = null,
        this.index = 0,
        this.ref = null,
```

```
    this.pendingProps = t,
    this.dependencies = this.memoizedState = this.updateQueue = this.memoizedProps = null,
    this.mode = r,
    this.effectTag = 0,
    this.lastEffect = this.firstEffect = this.nextEffect = null,
    this.childExpirationTime = this.expirationTime = 0,
    this.alternate = null
}
function Ru(e, t, n, r) {
    return new Ou(e,t,n,r)
}
function Iu(e) {
    return !(!(e = e.prototype) || !e.isReactComponent)
}
function Mu(e, t) {
    var n = e.alternate;
    return null === n ? ((n = Ru(e.tag, t, e.key, e.mode)).elementType = e.elementType,
    n.type = e.type,
    n.stateNode = e.stateNode,
    n.alternate = e,
    e.alternate = n) : (n.pendingProps = t,
    n.effectTag = 0,
    n.nextEffect = null,
    n.firstEffect = null,
    n.lastEffect = null),
    n.childExpirationTime = e.childExpirationTime,
    n.expirationTime = e.expirationTime,
    n.child = e.child,
    n.memoizedProps = e.memoizedProps,
    n.memoizedState = e.memoizedState,
    n.updateQueue = e.updateQueue,
    t = e.dependencies,
    n.dependencies = null === t ? null : {
        expirationTime: t.expirationTime,
        firstContext: t.firstContext,
        responders: t.responders
    },
    n.sibling = e.sibling,
    n.index = e.index,
    n.ref = e.ref,
    n
}
function Fu(e, t, n, r, l, i) {
    var o = 2;
    if (r = e,
    "function" === typeof e)
        Iu(e) && (o = 1);
    else if ("string" === typeof e)
        o = 5;
    else
        e: switch (e) {
        case ne:
            return Du(n.children, l, i, t);
        case oe:
            o = 8,
            l |= 7;
            break;
        case re:
            o = 8,
            l |= 1;
            break;
        case le:
            return (e = Ru(12, n, t, 8 | l)).elementType = le,
            e.type = le,
            e.expirationTime = i,
            e;
        case ce:
            return (e = Ru(13, n, t, l)).type = ce,
            e.elementType = ce,
```

```
                    e.expirationTime = i,
                    e;
                case se:
                    return (e = Ru(19, n, t, l)).elementType = se,
                    e.expirationTime = i,
                    e;
                default:
                    if ("object" === typeof e && null !== e)
                        switch (e.$$typeof) {
                        case ie:
                            o = 10;
                            break e;
                        case ae:
                            o = 9;
                            break e;
                        case ue:
                            o = 11;
                            break e;
                        case fe:
                            o = 14;
                            break e;
                        case de:
                            o = 16,
                            r = null;
                            break e;
                        case pe:
                            o = 22;
                            break e
                        }
                    throw Error(a(130, null == e ? e : typeof e, ""))
                }
            return (t = Ru(o, n, t, l)).elementType = e,
            t.type = r,
            t.expirationTime = i,
            t
        }
        function Du(e, t, n, r) {
            return (e = Ru(7, e, r, t)).expirationTime = n,
            e
        }
        function Lu(e, t, n) {
            return (e = Ru(6, e, null, t)).expirationTime = n,
            e
        }
        function Au(e, t, n) {
            return (t = Ru(4, null !== e.children ? e.children : [], e.key, t)).expirationTime = n,
            t.stateNode = {
                containerInfo: e.containerInfo,
                pendingChildren: null,
                implementation: e.implementation
            },
            t
        }
        function Uu(e, t, n) {
            this.tag = t,
            this.current = null,
            this.containerInfo = e,
            this.pingCache = this.pendingChildren = null,
            this.finishedExpirationTime = 0,
            this.finishedWork = null,
            this.timeoutHandle = -1,
            this.pendingContext = this.context = null,
            this.hydrate = n,
            this.callbackNode = null,
            this.callbackPriority = 90,
            this.lastExpiredTime = this.lastPingedTime = this.nextKnownPendingLevel = this.lastSuspendedTime =
this.firstSuspendedTime = this.firstPendingTime = 0
        }
        function Vu(e, t) {
```

```
            var n = e.firstSuspendedTime;
            return e = e.lastSuspendedTime,
            0 !== n && n >= t && e <= t
        }
        function Wu(e, t) {
            var n = e.firstSuspendedTime
              , r = e.lastSuspendedTime;
            n < t && (e.firstSuspendedTime = t),
            (r > t || 0 === n) && (e.lastSuspendedTime = t),
            t <= e.lastPingedTime && (e.lastPingedTime = 0),
            t <= e.lastExpiredTime && (e.lastExpiredTime = 0)
        }
        function ju(e, t) {
            t > e.firstPendingTime && (e.firstPendingTime = t);
            var n = e.firstSuspendedTime;
            0 !== n && (t >= n ? e.firstSuspendedTime = e.lastSuspendedTime = e.nextKnownPendingLevel = 0 : t >=
e.lastSuspendedTime && (e.lastSuspendedTime = t + 1),
            t > e.nextKnownPendingLevel && (e.nextKnownPendingLevel = t))
        }
        function Qu(e, t) {
            var n = e.lastExpiredTime;
            (0 === n || n > t) && (e.lastExpiredTime = t)
        }
        function $u(e, t, n, r) {
            var l = t.current
              , i = eu()
              , o = vi.suspense;
            i = tu(i, l, o);
            e: if (n) {
                t: {
                    if (et(n = n._reactInternalFiber) !== n || 1 !== n.tag)
                        throw Error(a(170));
                    var u = n;
                    do {
                        switch (u.tag) {
                        case 3:
                            u = u.stateNode.context;
                            break t;
                        case 1:
                            if (bl(u.type)) {
                                u = u.stateNode.__reactInternalMemoizedMergedChildContext;
                                break t
                            }
                        }
                        u = u.return
                    } while (null !== u);
                    throw Error(a(171))
                }
                if (1 === n.tag) {
                    var c = n.type;
                    if (bl(c)) {
                        n = xl(n, c, u);
                        break e
                    }
                }
                n = u
            } else
                n = ml;
            return null === t.context ? t.context = n : t.pendingContext = n,
            (t = fi(i, o)).payload = {
                element: e
            },
            null !== (r = void 0 === r ? null : r) && (t.callback = r),
            di(l, t),
            nu(l, i),
            i
        }
        function Bu(e) {
            if (!(e = e.current).child)
```

```
                return null;
            switch (e.child.tag) {
            case 5:
            default:
                return e.child.stateNode
            }
        }
        function Hu(e, t) {
            null !== (e = e.memoizedState) && null !== e.dehydrated && e.retryTime < t && (e.retryTime = t)
        }
        function Ku(e, t) {
            Hu(e, t),
            (e = e.alternate) && Hu(e, t)
        }
        function qu(e, t, n) {
            var r = new Uu(e,t,n = null != n && !0 === n.hydrate)
              , l = Ru(3, null, null, 2 === t ? 7 : 1 === t ? 3 : 0);
            r.current = l,
            l.stateNode = r,
            ci(l),
            e[zn] = r.current,
            n && 0 !== t && function(e, t) {
                var n = Je(t);
                _t.forEach((function(e) {
                    ht(e, t, n)
                }
                )),
                Pt.forEach((function(e) {
                    ht(e, t, n)
                }
                ))
            }(0, 9 === e.nodeType ? e : e.ownerDocument),
            this._internalRoot = r
        }
        function Yu(e) {
            return !(!e || 1 !== e.nodeType && 9 !== e.nodeType && 11 !== e.nodeType && (8 !== e.nodeType || "
react-mount-point-unstable " !== e.nodeValue))
        }
        function Xu(e, t, n, r, l) {
            var i = n._reactRootContainer;
            if (i) {
                var a = i._internalRoot;
                if ("function" === typeof l) {
                    var o = l;
                    l = function() {
                        var e = Bu(a);
                        o.call(e)
                    }
                }
                $u(t, a, e, l)
            } else {
                if (i = n._reactRootContainer = function(e, t) {
                    if (t || (t = !(!(t = e ? 9 === e.nodeType ? e.documentElement : e.firstChild : null) || 1 !==
t.nodeType || !t.hasAttribute("data-reactroot"))),
                    !t)
                        for (var n; n = e.lastChild; )
                            e.removeChild(n);
                    return new qu(e,0,t ? {
                        hydrate: !0
                    } : void 0)
                }(n, r),
                a = i._internalRoot,
                "function" === typeof l) {
                    var u = l;
                    l = function() {
                        var e = Bu(a);
                        u.call(e)
                    }
                }
```

```
        cu((function() {
            $u(t, a, e, l)
        }
        ))
    }
    return Bu(a)
}
function Gu(e, t, n) {
    var r = 3 < arguments.length && void 0 !== arguments[3] ? arguments[3] : null;
    return {
        $$typeof: te,
        key: null == r ? null : "" + r,
        children: e,
        containerInfo: t,
        implementation: n
    }
}
function Zu(e, t) {
    var n = 2 < arguments.length && void 0 !== arguments[2] ? arguments[2] : null;
    if (!Yu(t))
        throw Error(a(200));
    return Gu(e, t, null, n)
}
qu.prototype.render = function(e) {
    $u(e, this._internalRoot, null, null)
}
,
qu.prototype.unmount = function() {
    var e = this._internalRoot
      , t = e.containerInfo;
    $u(null, e, null, (function() {
        t[zn] = null
    }
    ))
}
,
vt = function(e) {
    if (13 === e.tag) {
        var t = Gl(eu(), 150, 100);
        nu(e, t),
        Ku(e, t)
    }
}
,
yt = function(e) {
    13 === e.tag && (nu(e, 3),
    Ku(e, 3))
}
,
gt = function(e) {
    if (13 === e.tag) {
        var t = eu();
        nu(e, t = tu(t, e, null)),
        Ku(e, t)
    }
}
,
P = function(e, t, n) {
    switch (t) {
    case "input":
        if (Se(e, n),
        t = n.name,
        "radio" === n.type && null != t) {
            for (n = e; n.parentNode; )
                n = n.parentNode;
            for (n = n.querySelectorAll("input[name=" + JSON.stringify("" + t) + '][type="radio"]'),
            t = 0; t < n.length; t++) {
                var r = n[t];
                if (r !== e && r.form === e.form) {
```

```
                    var l = Mn(r);
                    if (!l)
                        throw Error(a(90));
                    ke(r),
                    Se(r, l)
                }
            }
        }
        break;
    case "textarea":
        Re(e, n);
        break;
    case "select":
        null != (t = n.value) && Ne(e, !!n.multiple, t, !1)
    }
}
,
M = uu,
F = function(e, t, n, r, l) {
    var i = Oo;
    Oo |= 4;
    try {
        return Hl(98, e.bind(null, t, n, r, l))
    } finally {
        0 === (Oo = i) && Yl()
    }
}
,
D = function() {
    0 === (49 & Oo) && (function() {
        if (null !== Xo) {
            var e = Xo;
            Xo = null,
            e.forEach((function(e, t) {
                Qu(t, e),
                iu(t)
            }
            )),
            Yl()
        }
    }(),
    Tu())
}
,
L = function(e, t) {
    var n = Oo;
    Oo |= 2;
    try {
        return e(t)
    } finally {
        0 === (Oo = n) && Yl()
    }
}
;
var Ju = {
    Events: [Rn, In, Mn, C, T, Wn, function(e) {
        it(e, Vn)
    }
    , R, I, Zt, ut, Tu, {
        current: !1
    }]
};
!function(e) {
    var t = e.findFiberByHostInstance;
    (function(e) {
        if ("undefined" === typeof __REACT_DEVTOOLS_GLOBAL_HOOK__)
            return !1;
        var t = __REACT_DEVTOOLS_GLOBAL_HOOK__;
        if (t.isDisabled || !t.supportsFiber)
```

```
            return !0;
        try {
            var n = t.inject(e);
            Nu = function(e) {
                try {
                    t.onCommitFiberRoot(n, e, void 0, 64 === (64 & e.current.effectTag))
                } catch (r) {}
            }
            ,
            zu = function(e) {
                try {
                    t.onCommitFiberUnmount(n, e)
                } catch (r) {}
            }
        } catch (r) {}
    }
    )(l({}, e, {
        overrideHookState: null,
        overrideProps: null,
        setSuspenseHandler: null,
        scheduleUpdate: null,
        currentDispatcherRef: X.ReactCurrentDispatcher,
        findHostInstanceByFiber: function(e) {
            return null === (e = rt(e)) ? null : e.stateNode
        },
        findFiberByHostInstance: function(e) {
            return t ? t(e) : null
        },
        findHostInstancesForRefresh: null,
        scheduleRefresh: null,
        scheduleRoot: null,
        setRefreshHandler: null,
        getCurrentFiber: null
    }))
}({
    findFiberByHostInstance: On,
    bundleType: 0,
    version: "16.14.0",
    rendererPackageName: "react-dom"
}),
t.__SECRET_INTERNALS_DO_NOT_USE_OR_YOU_WILL_BE_FIRED = Ju,
t.createPortal = Zu,
t.findDOMNode = function(e) {
    if (null == e)
        return null;
    if (1 === e.nodeType)
        return e;
    var t = e._reactInternalFiber;
    if (void 0 === t) {
        if ("function" === typeof e.render)
            throw Error(a(188));
        throw Error(a(268, Object.keys(e)))
    }
    return e = null === (e = rt(t)) ? null : e.stateNode
}
,
t.flushSync = function(e, t) {
    if (0 !== (48 & Oo))
        throw Error(a(187));
    var n = Oo;
    Oo |= 1;
    try {
        return Hl(99, e.bind(null, t))
    } finally {
        Oo = n,
        Yl()
    }
}
,
```

```
        t.hydrate = function(e, t, n) {
            if (!Yu(t))
                throw Error(a(200));
            return Xu(null, e, t, !0, n)
        }
        ,
        t.render = function(e, t, n) {
            if (!Yu(t))
                throw Error(a(200));
            return Xu(null, e, t, !1, n)
        }
        ,
        t.unmountComponentAtNode = function(e) {
            if (!Yu(e))
                throw Error(a(40));
            return !!e._reactRootContainer && (cu((function() {
                Xu(null, null, e, !1, (function() {
                    e._reactRootContainer = null,
                    e[zn] = null
                }
                ))
            }
            )),
            !0)
        }
        ,
        t.unstable_batchedUpdates = uu,
        t.unstable_createPortal = function(e, t) {
            return Zu(e, t, 2 < arguments.length && void 0 !== arguments[2] ? arguments[2] : null)
        }
        ,
        t.unstable_renderSubtreeIntoContainer = function(e, t, n, r) {
            if (!Yu(n))
                throw Error(a(200));
            if (null == e || void 0 === e._reactInternalFiber)
                throw Error(a(38));
            return Xu(e, t, n, !1, r)
        }
        ,
        t.version = "16.14.0"
    },
    3935: function(e, t, n) {
        "use strict";
        !function e() {
            if ("undefined" !== typeof __REACT_DEVTOOLS_GLOBAL_HOOK__ && "function" === typeof
__REACT_DEVTOOLS_GLOBAL_HOOK__.checkDCE)
                try {
                    __REACT_DEVTOOLS_GLOBAL_HOOK__.checkDCE(e)
                } catch (t) {
                    console.error(t)
                }
        }(),
        e.exports = n(4448)
    },
    4203: function(e, t) {
        "use strict";
        var n, r, l, i, a;
        if ("undefined" === typeof window || "function" !== typeof MessageChannel) {
            var o = null
              , u = null
              , c = function() {
                if (null !== o)
                    try {
                        var e = t.unstable_now();
                        o(!0, e),
                        o = null
                    } catch (n) {
                        throw setTimeout(c, 0),
                        n
```

```
                }
            }
              , s = Date.now();
            t.unstable_now = function() {
                return Date.now() - s
            }
            ,
            n = function(e) {
                null !== o ? setTimeout(n, 0, e) : (o = e,
                setTimeout(c, 0))
            }
            ,
            r = function(e, t) {
                u = setTimeout(e, t)
            }
            ,
            l = function() {
                clearTimeout(u)
            }
            ,
            i = function() {
                return !1
            }
            ,
            a = t.unstable_forceFrameRate = function() {}
        } else {
            var f = window.performance
              , d = window.Date
              , p = window.setTimeout
              , m = window.clearTimeout;
            if ("undefined" !== typeof console) {
                var h = window.cancelAnimationFrame;
                "function" !== typeof window.requestAnimationFrame && console.error("This browser doesn't support
requestAnimationFrame. Make sure that you load a polyfill in older browsers. https://fb.me/react-polyfills"),
                "function" !== typeof h && console.error("This browser doesn't support cancelAnimationFrame. Make
sure that you load a polyfill in older browsers. https://fb.me/react-polyfills")
            }
            if ("object" === typeof f && "function" === typeof f.now)
                t.unstable_now = function() {
                    return f.now()
                }
                ;
            else {
                var v = d.now();
                t.unstable_now = function() {
                    return d.now() - v
                }
            }
            var y = !1
              , g = null
              , b = -1
              , w = 5
              , k = 0;
            i = function() {
                return t.unstable_now() >= k
            }
            ,
            a = function() {}
            ,
            t.unstable_forceFrameRate = function(e) {
                0 > e || 125 < e ? console.error("forceFrameRate takes a positive int between 0 and 125, forcing
framerates higher than 125 fps is not unsupported") : w = 0 < e ? Math.floor(1e3 / e) : 5
            }
            ;
            var x = new MessageChannel
              , T = x.port2;
            x.port1.onmessage = function() {
                if (null !== g) {
                    var e = t.unstable_now();
```

```
            k = e + w;
            try {
                g(!0, e) ? T.postMessage(null) : (y = !1,
                g = null)
            } catch (n) {
                throw T.postMessage(null),
                n
            }
        } else
            y = !1
    }
    ,
    n = function(e) {
        g = e,
        y || (y = !0,
        T.postMessage(null))
    }
    ,
    r = function(e, n) {
        b = p((function() {
            e(t.unstable_now())
        }
        ), n)
    }
    ,
    l = function() {
        m(b),
        b = -1
    }
}
function E(e, t) {
    var n = e.length;
    e.push(t);
    e: for (; ; ) {
        var r = n - 1 >>> 1
          , l = e[r];
        if (!(void 0 !== l && 0 < _(l, t)))
            break e;
        e[r] = t,
        e[n] = l,
        n = r
    }
}
function S(e) {
    return void 0 === (e = e[0]) ? null : e
}
function C(e) {
    var t = e[0];
    if (void 0 !== t) {
        var n = e.pop();
        if (n !== t) {
            e[0] = n;
            e: for (var r = 0, l = e.length; r < l; ) {
                var i = 2 * (r + 1) - 1
                  , a = e[i]
                  , o = i + 1
                  , u = e[o];
                if (void 0 !== a && 0 > _(a, n))
                    void 0 !== u && 0 > _(u, a) ? (e[r] = u,
                    e[o] = n,
                    r = o) : (e[r] = a,
                    e[i] = n,
                    r = i);
                else {
                    if (!(void 0 !== u && 0 > _(u, n)))
                        break e;
                    e[r] = u,
                    e[o] = n,
                    r = o
```

```
                }
            }
        }
        return t
    }
    return null
}
function _(e, t) {
    var n = e.sortIndex - t.sortIndex;
    return 0 !== n ? n : e.id - t.id
}
var P = []
  , N = []
  , z = 1
  , O = null
  , R = 3
  , I = !1
  , M = !1
  , F = !1;
function D(e) {
    for (var t = S(N); null !== t; ) {
        if (null === t.callback)
            C(N);
        else {
            if (!(t.startTime <= e))
                break;
            C(N),
            t.sortIndex = t.expirationTime,
            E(P, t)
        }
        t = S(N)
    }
}
function L(e) {
    if (F = !1,
    D(e),
    !M)
        if (null !== S(P))
            M = !0,
            n(A);
        else {
            var t = S(N);
            null !== t && r(L, t.startTime - e)
        }
}
function A(e, n) {
    M = !1,
    F && (F = !1,
    l()),
    I = !0;
    var a = R;
    try {
        for (D(n),
        O = S(P); null !== O && (!(O.expirationTime > n) || e && !i()); ) {
            var o = O.callback;
            if (null !== o) {
                O.callback = null,
                R = O.priorityLevel;
                var u = o(O.expirationTime <= n);
                n = t.unstable_now(),
                "function" === typeof u ? O.callback = u : O === S(P) && C(P),
                D(n)
            } else
                C(P);
            O = S(P)
        }
        if (null !== O)
            var c = !0;
        else {
```

```
            var s = S(N);
            null !== s && r(L, s.startTime - n),
            c = !1
        }
        return c
    } finally {
        O = null,
        R = a,
        I = !1
    }
}
function U(e) {
    switch (e) {
    case 1:
        return -1;
    case 2:
        return 250;
    case 5:
        return 1073741823;
    case 4:
        return 1e4;
    default:
        return 5e3
    }
}
var V = a;
t.unstable_IdlePriority = 5,
t.unstable_ImmediatePriority = 1,
t.unstable_LowPriority = 4,
t.unstable_NormalPriority = 3,
t.unstable_Profiling = null,
t.unstable_UserBlockingPriority = 2,
t.unstable_cancelCallback = function(e) {
    e.callback = null
}
,
t.unstable_continueExecution = function() {
    M || I || (M = !0,
    n(A))
}
,
t.unstable_getCurrentPriorityLevel = function() {
    return R
}
,
t.unstable_getFirstCallbackNode = function() {
    return S(P)
}
,
t.unstable_next = function(e) {
    switch (R) {
    case 1:
    case 2:
    case 3:
        var t = 3;
        break;
    default:
        t = R
    }
    var n = R;
    R = t;
    try {
        return e()
    } finally {
        R = n
    }
}
,
t.unstable_pauseExecution = function() {}
```

```
        ,
        t.unstable_requestPaint = V,
        t.unstable_runWithPriority = function(e, t) {
            switch (e) {
            case 1:
            case 2:
            case 3:
            case 4:
            case 5:
                break;
            default:
                e = 3
            }
            var n = R;
            R = e;
            try {
                return t()
            } finally {
                R = n
            }
        }
        ,
        t.unstable_scheduleCallback = function(e, i, a) {
            var o = t.unstable_now();
            if ("object" === typeof a && null !== a) {
                var u = a.delay;
                u = "number" === typeof u && 0 < u ? o + u : o,
                a = "number" === typeof a.timeout ? a.timeout : U(e)
            } else
                a = U(e),
                u = o;
            return e = {
                id: z++,
                callback: i,
                priorityLevel: e,
                startTime: u,
                expirationTime: a = u + a,
                sortIndex: -1
            },
            u > o ? (e.sortIndex = u,
            E(N, e),
            null === S(P) && e === S(N) && (F ? l() : F = !0,
            r(L, u - o))) : (e.sortIndex = a,
            E(P, e),
            M || I || (M = !0,
            n(A))),
            e
        }
        ,
        t.unstable_shouldYield = function() {
            var e = t.unstable_now();
            D(e);
            var n = S(P);
            return n !== O && null !== O && null !== n && null !== n.callback && n.startTime <= e &&
n.expirationTime < O.expirationTime || i()
        }
        ,
        t.unstable_wrapCallback = function(e) {
            var t = R;
            return function() {
                var n = R;
                R = t;
                try {
                    return e.apply(this, arguments)
                } finally {
                    R = n
                }
            }
        }
```

```
    },
    4142: function(e, t, n) {
        "use strict";
        e.exports = n(4203)
    },
    5251: function(e, t, n) {
        "use strict";
        var r = n(3027)
          , l = 60103;
        if (60107,
        "function" === typeof Symbol && Symbol.for) {
            var i = Symbol.for;
            l = i("react.element"),
            i("react.fragment")
        }
        var a = r.__SECRET_INTERNALS_DO_NOT_USE_OR_YOU_WILL_BE_FIRED.ReactCurrentOwner
          , o = Object.prototype.hasOwnProperty
          , u = {
            key: !0,
            ref: !0,
            __self: !0,
            __source: !0
        };
        function c(e, t, n) {
            var r, i = {}, c = null, s = null;
            for (r in void 0 !== n && (c = "" + n),
            void 0 !== t.key && (c = "" + t.key),
            void 0 !== t.ref && (s = t.ref),
            t)
                o.call(t, r) && !u.hasOwnProperty(r) && (i[r] = t[r]);
            if (e && e.defaultProps)
                for (r in t = e.defaultProps)
                    void 0 === i[r] && (i[r] = t[r]);
            return {
                $$typeof: l,
                type: e,
                key: c,
                ref: s,
                props: i,
                _owner: a.current
            }
        }
        t.jsx = c,
        t.jsxs = c
    },
    2408: function(e, t, n) {
        "use strict";
        var r = n(6086)
          , l = "function" === typeof Symbol && Symbol.for
          , i = l ? Symbol.for("react.element") : 60103
          , a = l ? Symbol.for("react.portal") : 60106
          , o = l ? Symbol.for("react.fragment") : 60107
          , u = l ? Symbol.for("react.strict_mode") : 60108
          , c = l ? Symbol.for("react.profiler") : 60114
          , s = l ? Symbol.for("react.provider") : 60109
          , f = l ? Symbol.for("react.context") : 60110
          , d = l ? Symbol.for("react.forward_ref") : 60112
          , p = l ? Symbol.for("react.suspense") : 60113
          , m = l ? Symbol.for("react.memo") : 60115
          , h = l ? Symbol.for("react.lazy") : 60116
          , v = "function" === typeof Symbol && Symbol.iterator;
        function y(e) {
            for (var t = "https://reactjs.org/docs/error-decoder.html?invariant=" + e, n = 1; n < arguments.length;
n++)
                t += "&args[]=" + encodeURIComponent(arguments[n]);
            return "Minified React error #" + e + "; visit " + t + " for the full message or use the non-minified
dev environment for full errors and additional helpful warnings."
        }
        var g = {
```

```
    isMounted: function() {
        return !1
    },
    enqueueForceUpdate: function() {},
    enqueueReplaceState: function() {},
    enqueueSetState: function() {}
}
  , b = {};
function w(e, t, n) {
    this.props = e,
    this.context = t,
    this.refs = b,
    this.updater = n || g
}
function k() {}
function x(e, t, n) {
    this.props = e,
    this.context = t,
    this.refs = b,
    this.updater = n || g
}
w.prototype.isReactComponent = {},
w.prototype.setState = function(e, t) {
    if ("object" !== typeof e && "function" !== typeof e && null != e)
        throw Error(y(85));
    this.updater.enqueueSetState(this, e, t, "setState")
}
,
w.prototype.forceUpdate = function(e) {
    this.updater.enqueueForceUpdate(this, e, "forceUpdate")
}
,
k.prototype = w.prototype;
var T = x.prototype = new k;
T.constructor = x,
r(T, w.prototype),
T.isPureReactComponent = !0;
var E = {
    current: null
}
  , S = Object.prototype.hasOwnProperty
  , C = {
    key: !0,
    ref: !0,
    __self: !0,
    __source: !0
};
function _(e, t, n) {
    var r, l = {}, a = null, o = null;
    if (null != t)
        for (r in void 0 !== t.ref && (o = t.ref),
        void 0 !== t.key && (a = "" + t.key),
        t)
            S.call(t, r) && !C.hasOwnProperty(r) && (l[r] = t[r]);
    var u = arguments.length - 2;
    if (1 === u)
        l.children = n;
    else if (1 < u) {
        for (var c = Array(u), s = 0; s < u; s++)
            c[s] = arguments[s + 2];
        l.children = c
    }
    if (e && e.defaultProps)
        for (r in u = e.defaultProps)
            void 0 === l[r] && (l[r] = u[r]);
    return {
        $$typeof: i,
        type: e,
        key: a,
```

```
            ref: o,
            props: l,
            _owner: E.current
        }
    }
    function P(e) {
        return "object" === typeof e && null !== e && e.$$typeof === i
    }
    var N = /\/+/g
      , z = [];
    function O(e, t, n, r) {
        if (z.length) {
            var l = z.pop();
            return l.result = e,
            l.keyPrefix = t,
            l.func = n,
            l.context = r,
            l.count = 0,
            l
        }
        return {
            result: e,
            keyPrefix: t,
            func: n,
            context: r,
            count: 0
        }
    }
    function R(e) {
        e.result = null,
        e.keyPrefix = null,
        e.func = null,
        e.context = null,
        e.count = 0,
        10 > z.length && z.push(e)
    }
    function I(e, t, n, r) {
        var l = typeof e;
        "undefined" !== l && "boolean" !== l || (e = null);
        var o = !1;
        if (null === e)
            o = !0;
        else
            switch (l) {
            case "string":
            case "number":
                o = !0;
                break;
            case "object":
                switch (e.$$typeof) {
                case i:
                case a:
                    o = !0
                }
            }
        if (o)
            return n(r, e, "" === t ? "." + F(e, 0) : t),
            1;
        if (o = 0,
        t = "" === t ? "." : t + ":",
        Array.isArray(e))
            for (var u = 0; u < e.length; u++) {
                var c = t + F(l = e[u], u);
                o += I(l, c, n, r)
            }
        else if (null === e || "object" !== typeof e ? c = null : c = "function" === typeof (c = v && e[v] ||
e["@@iterator"]) ? c : null,
        "function" === typeof c)
            for (e = c.call(e),
```

```
                u = 0; !(l = e.next()).done; )
                    o += I(l = l.value, c = t + F(l, u++), n, r);
            else if ("object" === l)
                throw n = "" + e,
                Error(y(31, "[object Object]" === n ? "object with keys {" + Object.keys(e).join(", ") + "}" : n,
""));
            return o
        }
        function M(e, t, n) {
            return null == e ? 0 : I(e, "", t, n)
        }
        function F(e, t) {
            return "object" === typeof e && null !== e && null != e.key ? function(e) {
                var t = {
                    "=": "=0",
                    ":": "=2"
                };
                return "$" + ("" + e).replace(/[=:]/g, (function(e) {
                    return t[e]
                }
                ))
            }(e.key) : t.toString(36)
        }
        function D(e, t) {
            e.func.call(e.context, t, e.count++)
        }
        function L(e, t, n) {
            var r = e.result
              , l = e.keyPrefix;
            e = e.func.call(e.context, t, e.count++),
            Array.isArray(e) ? A(e, r, n, (function(e) {
                return e
            }
            )) : null != e && (P(e) && (e = function(e, t) {
                return {
                    $$typeof: i,
                    type: e.type,
                    key: t,
                    ref: e.ref,
                    props: e.props,
                    _owner: e._owner
                }
            }(e, l + (!e.key || t && t.key === e.key ? "" : ("" + e.key).replace(N, "$&/") + "/") + n)),
            r.push(e))
        }
        function A(e, t, n, r, l) {
            var i = "";
            null != n && (i = ("" + n).replace(N, "$&/") + "/"),
            M(e, L, t = O(t, i, r, l)),
            R(t)
        }
        var U = {
            current: null
        };
        function V() {
            var e = U.current;
            if (null === e)
                throw Error(y(321));
            return e
        }
        var W = {
            ReactCurrentDispatcher: U,
            ReactCurrentBatchConfig: {
                suspense: null
            },
            ReactCurrentOwner: E,
            IsSomeRendererActing: {
                current: !1
            },
```

```
    assign: r
};
t.Children = {
    map: function(e, t, n) {
        if (null == e)
            return e;
        var r = [];
        return A(e, r, null, t, n),
        r
    },
    forEach: function(e, t, n) {
        if (null == e)
            return e;
        M(e, D, t = O(null, null, t, n)),
        R(t)
    },
    count: function(e) {
        return M(e, (function() {
            return null
        }
        ), null)
    },
    toArray: function(e) {
        var t = [];
        return A(e, t, null, (function(e) {
            return e
        }
        )),
        t
    },
    only: function(e) {
        if (!P(e))
            throw Error(y(143));
        return e
    }
},
t.Component = w,
t.Fragment = o,
t.Profiler = c,
t.PureComponent = x,
t.StrictMode = u,
t.Suspense = p,
t.__SECRET_INTERNALS_DO_NOT_USE_OR_YOU_WILL_BE_FIRED = W,
t.cloneElement = function(e, t, n) {
    if (null === e || void 0 === e)
        throw Error(y(267, e));
    var l = r({}, e.props)
      , a = e.key
      , o = e.ref
      , u = e._owner;
    if (null != t) {
        if (void 0 !== t.ref && (o = t.ref,
        u = E.current),
        void 0 !== t.key && (a = "" + t.key),
        e.type && e.type.defaultProps)
            var c = e.type.defaultProps;
        for (s in t)
            S.call(t, s) && !C.hasOwnProperty(s) && (l[s] = void 0 === t[s] && void 0 !== c ? c[s] : t[s])
    }
    var s = arguments.length - 2;
    if (1 === s)
        l.children = n;
    else if (1 < s) {
        c = Array(s);
        for (var f = 0; f < s; f++)
            c[f] = arguments[f + 2];
        l.children = c
    }
    return {
```

```
        $$typeof: i,
        type: e.type,
        key: a,
        ref: o,
        props: l,
        _owner: u
    }
}
,
t.createContext = function(e, t) {
    return void 0 === t && (t = null),
    (e = {
        $$typeof: f,
        _calculateChangedBits: t,
        _currentValue: e,
        _currentValue2: e,
        _threadCount: 0,
        Provider: null,
        Consumer: null
    }).Provider = {
        $$typeof: s,
        _context: e
    },
    e.Consumer = e
}
,
t.createElement = _,
t.createFactory = function(e) {
    var t = _.bind(null, e);
    return t.type = e,
    t
}
,
t.createRef = function() {
    return {
        current: null
    }
}
,
t.forwardRef = function(e) {
    return {
        $$typeof: d,
        render: e
    }
}
,
t.isValidElement = P,
t.lazy = function(e) {
    return {
        $$typeof: h,
        _ctor: e,
        _status: -1,
        _result: null
    }
}
,
t.memo = function(e, t) {
    return {
        $$typeof: m,
        type: e,
        compare: void 0 === t ? null : t
    }
}
,
t.useCallback = function(e, t) {
    return V().useCallback(e, t)
}
,
t.useContext = function(e, t) {
```

```
            return V().useContext(e, t)
        }
        ,
        t.useDebugValue = function() {}
        ,
        t.useEffect = function(e, t) {
            return V().useEffect(e, t)
        }
        ,
        t.useImperativeHandle = function(e, t, n) {
            return V().useImperativeHandle(e, t, n)
        }
        ,
        t.useLayoutEffect = function(e, t) {
            return V().useLayoutEffect(e, t)
        }
        ,
        t.useMemo = function(e, t) {
            return V().useMemo(e, t)
        }
        ,
        t.useReducer = function(e, t, n) {
            return V().useReducer(e, t, n)
        }
        ,
        t.useRef = function(e) {
            return V().useRef(e)
        }
        ,
        t.useState = function(e) {
            return V().useState(e)
        }
        ,
        t.version = "16.14.0"
    },
    7294: function(e, t, n) {
        "use strict";
        e.exports = n(2408)
    },
    5893: function(e, t, n) {
        "use strict";
        e.exports = n(5251)
    }
}]);
(function () {
    "use strict";

    var MessageTypeUItoBG;
    (function (MessageTypeUItoBG) {
        MessageTypeUItoBG["GET_DATA"] = "ui-bg-get-data";
        MessageTypeUItoBG["GET_DEVTOOLS_DATA"] = "ui-bg-get-devtools-data";
        MessageTypeUItoBG["SUBSCRIBE_TO_CHANGES"] =
            "ui-bg-subscribe-to-changes";
        MessageTypeUItoBG["UNSUBSCRIBE_FROM_CHANGES"] =
            "ui-bg-unsubscribe-from-changes";
        MessageTypeUItoBG["CHANGE_SETTINGS"] = "ui-bg-change-settings";
        MessageTypeUItoBG["SET_THEME"] = "ui-bg-set-theme";
        MessageTypeUItoBG["TOGGLE_ACTIVE_TAB"] = "ui-bg-toggle-active-tab";
        MessageTypeUItoBG["MARK_NEWS_AS_READ"] = "ui-bg-mark-news-as-read";
        MessageTypeUItoBG["MARK_NEWS_AS_DISPLAYED"] =
            "ui-bg-mark-news-as-displayed";
        MessageTypeUItoBG["LOAD_CONFIG"] = "ui-bg-load-config";
        MessageTypeUItoBG["APPLY_DEV_DYNAMIC_THEME_FIXES"] =
            "ui-bg-apply-dev-dynamic-theme-fixes";
        MessageTypeUItoBG["RESET_DEV_DYNAMIC_THEME_FIXES"] =
            "ui-bg-reset-dev-dynamic-theme-fixes";
        MessageTypeUItoBG["APPLY_DEV_INVERSION_FIXES"] =
            "ui-bg-apply-dev-inversion-fixes";
        MessageTypeUItoBG["RESET_DEV_INVERSION_FIXES"] =
```

```
        "ui-bg-reset-dev-inversion-fixes";
    MessageTypeUItoBG["APPLY_DEV_STATIC_THEMES"] =
        "ui-bg-apply-dev-static-themes";
    MessageTypeUItoBG["RESET_DEV_STATIC_THEMES"] =
        "ui-bg-reset-dev-static-themes";
    MessageTypeUItoBG["COLOR_SCHEME_CHANGE"] = "ui-bg-color-scheme-change";
    MessageTypeUItoBG["HIDE_HIGHLIGHTS"] = "ui-bg-hide-highlights";
})(MessageTypeUItoBG || (MessageTypeUItoBG = {}));
var MessageTypeBGtoUI;
(function (MessageTypeBGtoUI) {
    MessageTypeBGtoUI["CHANGES"] = "bg-ui-changes";
})(MessageTypeBGtoUI || (MessageTypeBGtoUI = {}));
var DebugMessageTypeBGtoUI;
(function (DebugMessageTypeBGtoUI) {
    DebugMessageTypeBGtoUI["CSS_UPDATE"] = "debug-bg-ui-css-update";
    DebugMessageTypeBGtoUI["UPDATE"] = "debug-bg-ui-update";
})(DebugMessageTypeBGtoUI || (DebugMessageTypeBGtoUI = {}));
var MessageTypeBGtoCS;
(function (MessageTypeBGtoCS) {
    MessageTypeBGtoCS["ADD_CSS_FILTER"] = "bg-cs-add-css-filter";
    MessageTypeBGtoCS["ADD_DYNAMIC_THEME"] = "bg-cs-add-dynamic-theme";
    MessageTypeBGtoCS["ADD_STATIC_THEME"] = "bg-cs-add-static-theme";
    MessageTypeBGtoCS["ADD_SVG_FILTER"] = "bg-cs-add-svg-filter";
    MessageTypeBGtoCS["CLEAN_UP"] = "bg-cs-clean-up";
    MessageTypeBGtoCS["FETCH_RESPONSE"] = "bg-cs-fetch-response";
    MessageTypeBGtoCS["UNSUPPORTED_SENDER"] = "bg-cs-unsupported-sender";
})(MessageTypeBGtoCS || (MessageTypeBGtoCS = {}));
var DebugMessageTypeBGtoCS;
(function (DebugMessageTypeBGtoCS) {
    DebugMessageTypeBGtoCS["RELOAD"] = "debug-bg-cs-reload";
})(DebugMessageTypeBGtoCS || (DebugMessageTypeBGtoCS = {}));
var MessageTypeCStoBG;
(function (MessageTypeCStoBG) {
    MessageTypeCStoBG["COLOR_SCHEME_CHANGE"] = "cs-bg-color-scheme-change";
    MessageTypeCStoBG["DARK_THEME_DETECTED"] = "cs-bg-dark-theme-detected";
    MessageTypeCStoBG["DARK_THEME_NOT_DETECTED"] =
        "cs-bg-dark-theme-not-detected";
    MessageTypeCStoBG["FETCH"] = "cs-bg-fetch";
    MessageTypeCStoBG["DOCUMENT_CONNECT"] = "cs-bg-document-connect";
    MessageTypeCStoBG["DOCUMENT_FORGET"] = "cs-bg-document-forget";
    MessageTypeCStoBG["DOCUMENT_FREEZE"] = "cs-bg-document-freeze";
    MessageTypeCStoBG["DOCUMENT_RESUME"] = "cs-bg-document-resume";
})(MessageTypeCStoBG || (MessageTypeCStoBG = {}));
var DebugMessageTypeCStoBG;
(function (DebugMessageTypeCStoBG) {
    DebugMessageTypeCStoBG["LOG"] = "debug-cs-bg-log";
})(DebugMessageTypeCStoBG || (DebugMessageTypeCStoBG = {}));
var MessageTypeCStoUI;
(function (MessageTypeCStoUI) {
    MessageTypeCStoUI["EXPORT_CSS_RESPONSE"] = "cs-ui-export-css-response";
})(MessageTypeCStoUI || (MessageTypeCStoUI = {}));
var MessageTypeUItoCS;
(function (MessageTypeUItoCS) {
    MessageTypeUItoCS["EXPORT_CSS"] = "ui-cs-export-css";
})(MessageTypeUItoCS || (MessageTypeUItoCS = {}));

function logInfo(...args) {}
function logWarn(...args) {}
function logInfoCollapsed(title, ...args) {}

function throttle(callback) {
    let pending = false;
    let frameId = null;
    let lastArgs;
    const throttled = (...args) => {
        lastArgs = args;
        if (frameId) {
            pending = true;
        } else {
```

```
            callback(...lastArgs);
            frameId = requestAnimationFrame(() => {
                frameId = null;
                if (pending) {
                    callback(...lastArgs);
                    pending = false;
                }
            });
        }
    };
    const cancel = () => {
        cancelAnimationFrame(frameId);
        pending = false;
        frameId = null;
    };
    return Object.assign(throttled, {cancel});
}
function createAsyncTasksQueue() {
    const tasks = [];
    let frameId = null;
    function runTasks() {
        let task;
        while ((task = tasks.shift())) {
            task();
        }
        frameId = null;
    }
    function add(task) {
        tasks.push(task);
        if (!frameId) {
            frameId = requestAnimationFrame(runTasks);
        }
    }
    function cancel() {
        tasks.splice(0);
        cancelAnimationFrame(frameId);
        frameId = null;
    }
    return {add, cancel};
}

function isArrayLike(items) {
    return items.length != null;
}
function forEach(items, iterator) {
    if (isArrayLike(items)) {
        for (let i = 0, len = items.length; i < len; i++) {
            iterator(items[i]);
        }
    } else {
        for (const item of items) {
            iterator(item);
        }
    }
}
function push(array, addition) {
    forEach(addition, (a) => array.push(a));
}
function toArray(items) {
    const results = [];
    for (let i = 0, len = items.length; i < len; i++) {
        results.push(items[i]);
    }
    return results;
}

function getDuration(time) {
    let duration = 0;
    if (time.seconds) {
```

```
        duration += time.seconds * 1000;
    }
    if (time.minutes) {
        duration += time.minutes * 60 * 1000;
    }
    if (time.hours) {
        duration += time.hours * 60 * 60 * 1000;
    }
    if (time.days) {
        duration += time.days * 24 * 60 * 60 * 1000;
    }
    return duration;
}

function createNodeAsap({
    selectNode,
    createNode,
    updateNode,
    selectTarget,
    createTarget,
    isTargetMutation
}) {
    const target = selectTarget();
    if (target) {
        const prev = selectNode();
        if (prev) {
            updateNode(prev);
        } else {
            createNode(target);
        }
    } else {
        const observer = new MutationObserver((mutations) => {
            const mutation = mutations.find(isTargetMutation);
            if (mutation) {
                unsubscribe();
                const target = selectTarget();
                selectNode() || createNode(target);
            }
        });
        const ready = () => {
            if (document.readyState !== "complete") {
                return;
            }
            unsubscribe();
            const target = selectTarget() || createTarget();
            selectNode() || createNode(target);
        };
        const unsubscribe = () => {
            document.removeEventListener("readystatechange", ready);
            observer.disconnect();
        };
        if (document.readyState === "complete") {
            ready();
        } else {
            document.addEventListener("readystatechange", ready);
            observer.observe(document, {childList: true, subtree: true});
        }
    }
}
function removeNode(node) {
    node && node.parentNode && node.parentNode.removeChild(node);
}
function watchForNodePosition(node, mode, onRestore = Function.prototype) {
    const MAX_ATTEMPTS_COUNT = 10;
    const RETRY_TIMEOUT = getDuration({seconds: 2});
    const ATTEMPTS_INTERVAL = getDuration({seconds: 10});
    const prevSibling = node.previousSibling;
    let parent = node.parentNode;
    if (!parent) {
```

```
    throw new Error(
        "Unable to watch for node position: parent element not found"
    );
}
if (mode === "prev-sibling" && !prevSibling) {
    throw new Error(
        "Unable to watch for node position: there is no previous sibling"
    );
}
let attempts = 0;
let start = null;
let timeoutId = null;
const restore = throttle(() => {
    if (timeoutId) {
        return;
    }
    attempts++;
    const now = Date.now();
    if (start == null) {
        start = now;
    } else if (attempts >= MAX_ATTEMPTS_COUNT) {
        if (now - start < ATTEMPTS_INTERVAL) {
            timeoutId = setTimeout(() => {
                start = null;
                attempts = 0;
                timeoutId = null;
                restore();
            }, RETRY_TIMEOUT);
            return;
        }
        start = now;
        attempts = 1;
    }
    if (mode === "head") {
        if (prevSibling && prevSibling.parentNode !== parent) {
            stop();
            return;
        }
    }
    if (mode === "prev-sibling") {
        if (prevSibling.parentNode == null) {
            stop();
            return;
        }
        if (prevSibling.parentNode !== parent) {
            updateParent(prevSibling.parentNode);
        }
    }
    if (mode === "head" && !parent.isConnected) {
        parent = document.head;
    }
    parent.insertBefore(
        node,
        prevSibling && prevSibling.isConnected
            ? prevSibling.nextSibling
            : parent.firstChild
    );
    observer.takeRecords();
    onRestore && onRestore();
});
const observer = new MutationObserver(() => {
    if (
        (mode === "head" &&
            (node.parentNode !== parent ||
                !node.parentNode.isConnected)) ||
        (mode === "prev-sibling" &&
            node.previousSibling !== prevSibling)
    ) {
        restore();
```

```
        }
    });
    const run = () => {
        observer.observe(parent, {childList: true});
    };
    const stop = () => {
        clearTimeout(timeoutId);
        observer.disconnect();
        restore.cancel();
    };
    const skip = () => {
        observer.takeRecords();
    };
    const updateParent = (parentNode) => {
        parent = parentNode;
        stop();
        run();
    };
    run();
    return {run, stop, skip};
}
function iterateShadowHosts(root, iterator) {
    if (root == null) {
        return;
    }
    const walker = document.createTreeWalker(
        root,
        NodeFilter.SHOW_ELEMENT,
        {
            acceptNode(node) {
                return node.shadowRoot == null
                    ? NodeFilter.FILTER_SKIP
                    : NodeFilter.FILTER_ACCEPT;
            }
        }
    );
    for (
        let node = root.shadowRoot ? walker.currentNode : walker.nextNode();
        node != null;
        node = walker.nextNode()
    ) {
        if (node.classList.contains("surfingkeys_hints_host")) {
            continue;
        }
        iterator(node);
        iterateShadowHosts(node.shadowRoot, iterator);
    }
}
let isDOMReady = () => {
    return (
        document.readyState === "complete" ||
        document.readyState === "interactive"
    );
};
function setIsDOMReady(newFunc) {
    isDOMReady = newFunc;
}
const readyStateListeners = new Set();
function addDOMReadyListener(listener) {
    isDOMReady() ? listener() : readyStateListeners.add(listener);
}
function removeDOMReadyListener(listener) {
    readyStateListeners.delete(listener);
}
function isReadyStateComplete() {
    return document.readyState === "complete";
}
const readyStateCompleteListeners = new Set();
function addReadyStateCompleteListener(listener) {
```

```
    isReadyStateComplete()
        ? listener()
        : readyStateCompleteListeners.add(listener);
}
function cleanReadyStateCompleteListeners() {
    readyStateCompleteListeners.clear();
}
if (!isDOMReady()) {
    const onReadyStateChange = () => {
        if (isDOMReady()) {
            readyStateListeners.forEach((listener) => listener());
            readyStateListeners.clear();
            if (isReadyStateComplete()) {
                document.removeEventListener(
                    "readystatechange",
                    onReadyStateChange
                );
                readyStateCompleteListeners.forEach((listener) =>
                    listener()
                );
                readyStateCompleteListeners.clear();
            }
        }
    };
    document.addEventListener("readystatechange", onReadyStateChange);
}
const HUGE_MUTATIONS_COUNT = 1000;
function isHugeMutation(mutations) {
    if (mutations.length > HUGE_MUTATIONS_COUNT) {
        return true;
    }
    let addedNodesCount = 0;
    for (let i = 0; i < mutations.length; i++) {
        addedNodesCount += mutations[i].addedNodes.length;
        if (addedNodesCount > HUGE_MUTATIONS_COUNT) {
            return true;
        }
    }
    return false;
}
function getElementsTreeOperations(mutations) {
    const additions = new Set();
    const deletions = new Set();
    const moves = new Set();
    mutations.forEach((m) => {
        forEach(m.addedNodes, (n) => {
            if (n instanceof Element && n.isConnected) {
                additions.add(n);
            }
        });
        forEach(m.removedNodes, (n) => {
            if (n instanceof Element) {
                if (n.isConnected) {
                    moves.add(n);
                    additions.delete(n);
                } else {
                    deletions.add(n);
                }
            }
        });
    });
    const duplicateAdditions = [];
    const duplicateDeletions = [];
    additions.forEach((node) => {
        if (additions.has(node.parentElement)) {
            duplicateAdditions.push(node);
        }
    });
    deletions.forEach((node) => {
```

```
        if (deletions.has(node.parentElement)) {
            duplicateDeletions.push(node);
        }
    });
    duplicateAdditions.forEach((node) => additions.delete(node));
    duplicateDeletions.forEach((node) => deletions.delete(node));
    return {additions, moves, deletions};
}
const optimizedTreeObservers = new Map();
const optimizedTreeCallbacks = new WeakMap();
function createOptimizedTreeObserver(root, callbacks) {
    let observer;
    let observerCallbacks;
    let domReadyListener;
    if (optimizedTreeObservers.has(root)) {
        observer = optimizedTreeObservers.get(root);
        observerCallbacks = optimizedTreeCallbacks.get(observer);
    } else {
        let hadHugeMutationsBefore = false;
        let subscribedForReadyState = false;
        observer = new MutationObserver((mutations) => {
            if (isHugeMutation(mutations)) {
                if (!hadHugeMutationsBefore || isDOMReady()) {
                    observerCallbacks.forEach(({onHugeMutations}) =>
                        onHugeMutations(root)
                    );
                } else if (!subscribedForReadyState) {
                    domReadyListener = () =>
                        observerCallbacks.forEach(({onHugeMutations}) =>
                            onHugeMutations(root)
                        );
                    addDOMReadyListener(domReadyListener);
                    subscribedForReadyState = true;
                }
                hadHugeMutationsBefore = true;
            } else {
                const elementsOperations =
                    getElementsTreeOperations(mutations);
                observerCallbacks.forEach(({onMinorMutations}) =>
                    onMinorMutations(elementsOperations)
                );
            }
        });
        observer.observe(root, {childList: true, subtree: true});
        optimizedTreeObservers.set(root, observer);
        observerCallbacks = new Set();
        optimizedTreeCallbacks.set(observer, observerCallbacks);
    }
    observerCallbacks.add(callbacks);
    return {
        disconnect() {
            observerCallbacks.delete(callbacks);
            if (domReadyListener) {
                removeDOMReadyListener(domReadyListener);
            }
            if (observerCallbacks.size === 0) {
                observer.disconnect();
                optimizedTreeCallbacks.delete(observer);
                optimizedTreeObservers.delete(root);
            }
        }
    };
}

function createOrUpdateStyle$1(css, type) {
    createNodeAsap({
        selectNode: () => document.getElementById("dark-reader-style"),
        createNode: (target) => {
            document.documentElement.setAttribute(
```

```
                "data-darkreader-mode",
                type
            );
            const style = document.createElement("style");
            style.id = "dark-reader-style";
            style.classList.add("darkreader");
            style.type = "text/css";
            style.textContent = css;
            target.appendChild(style);
        },
        updateNode: (existing) => {
            if (
                css.replace(/^\s+/gm, "") !==
                existing.textContent.replace(/^\s+/gm, "")
            ) {
                existing.textContent = css;
            }
        },
        selectTarget: () => document.head,
        createTarget: () => {
            const head = document.createElement("head");
            document.documentElement.insertBefore(
                head,
                document.documentElement.firstElementChild
            );
            return head;
        },
        isTargetMutation: (mutation) =>
            mutation.target.nodeName.toLowerCase() === "head"
    });
}
function removeStyle() {
    removeNode(document.getElementById("dark-reader-style"));
    document.documentElement.removeAttribute("data-darkreader-mode");
}

function createOrUpdateSVGFilter(svgMatrix, svgReverseMatrix) {
    createNodeAsap({
        selectNode: () => document.getElementById("dark-reader-svg"),
        createNode: (target) => {
            const SVG_NS = "http://www.w3.org/2000/svg";
            const createMatrixFilter = (id, matrix) => {
                const filter = document.createElementNS(SVG_NS, "filter");
                filter.id = id;
                filter.style.colorInterpolationFilters = "sRGB";
                filter.setAttribute("x", "0");
                filter.setAttribute("y", "0");
                filter.setAttribute("width", "99999");
                filter.setAttribute("height", "99999");
                filter.appendChild(createColorMatrix(matrix));
                return filter;
            };
            const createColorMatrix = (matrix) => {
                const colorMatrix = document.createElementNS(
                    SVG_NS,
                    "feColorMatrix"
                );
                colorMatrix.setAttribute("type", "matrix");
                colorMatrix.setAttribute("values", matrix);
                return colorMatrix;
            };
            const svg = document.createElementNS(SVG_NS, "svg");
            svg.id = "dark-reader-svg";
            svg.style.height = "0";
            svg.style.width = "0";
            svg.appendChild(
                createMatrixFilter("dark-reader-filter", svgMatrix)
            );
            svg.appendChild(
```

```
                createMatrixFilter(
                    "dark-reader-reverse-filter",
                    svgReverseMatrix
                )
            );
            target.appendChild(svg);
        },
        updateNode: (existing) => {
            const existingMatrix = existing.firstChild.firstChild;
            if (existingMatrix.getAttribute("values") !== svgMatrix) {
                existingMatrix.setAttribute("values", svgMatrix);
                const style = document.getElementById("dark-reader-style");
                const css = style.textContent;
                style.textContent = "";
                style.textContent = css;
            }
        },
        selectTarget: () => document.head,
        createTarget: () => {
            const head = document.createElement("head");
            document.documentElement.insertBefore(
                head,
                document.documentElement.firstElementChild
            );
            return head;
        },
        isTargetMutation: (mutation) =>
            mutation.target.nodeName.toLowerCase() === "head"
    });
}
function removeSVGFilter() {
    removeNode(document.getElementById("dark-reader-svg"));
}

function evalMath(expression) {
    const rpnStack = [];
    const workingStack = [];
    let lastToken;
    for (let i = 0, len = expression.length; i < len; i++) {
        const token = expression[i];
        if (!token || token === " ") {
            continue;
        }
        if (operators.has(token)) {
            const op = operators.get(token);
            while (workingStack.length) {
                const currentOp = operators.get(workingStack[0]);
                if (!currentOp) {
                    break;
                }
                if (op.lessOrEqualThan(currentOp)) {
                    rpnStack.push(workingStack.shift());
                } else {
                    break;
                }
            }
            workingStack.unshift(token);
        } else if (!lastToken || operators.has(lastToken)) {
            rpnStack.push(token);
        } else {
            rpnStack[rpnStack.length - 1] += token;
        }
        lastToken = token;
    }
    rpnStack.push(...workingStack);
    const stack = [];
    for (let i = 0, len = rpnStack.length; i < len; i++) {
        const op = operators.get(rpnStack[i]);
        if (op) {
```

```
            const args = stack.splice(0, 2);
            stack.push(op.exec(args[1], args[0]));
        } else {
            stack.unshift(parseFloat(rpnStack[i]));
        }
    }
    return stack[0];
}
class Operator {
    constructor(precedence, method) {
        this.precendce = precedence;
        this.execMethod = method;
    }
    exec(left, right) {
        return this.execMethod(left, right);
    }
    lessOrEqualThan(op) {
        return this.precendce <= op.precendce;
    }
}
const operators = new Map([
    ["+", new Operator(1, (left, right) => left + right)],
    ["-", new Operator(1, (left, right) => left - right)],
    ["*", new Operator(2, (left, right) => left * right)],
    ["/", new Operator(2, (left, right) => left / right)]
]);

function getMatches(regex, input, group = 0) {
    const matches = [];
    let m;
    while ((m = regex.exec(input))) {
        matches.push(m[group]);
    }
    return matches;
}
function formatCSS(text) {
    function trimLeft(text) {
        return text.replace(/^\s+/, "");
    }
    function getIndent(depth) {
        if (depth === 0) {
            return "";
        }
        return " ".repeat(4 * depth);
    }
    if (text.length < 50000) {
        const emptyRuleRegexp = /[^{}]+{\s*}/;
        while (emptyRuleRegexp.test(text)) {
            text = text.replace(emptyRuleRegexp, "");
        }
    }
    const css = text
        .replace(/\s{2,}/g, " ")
        .replace(/\{/g, "{\n")
        .replace(/\}/g, "\n}\n")
        .replace(/\;(?![^\(|\"]*(\)|\"))/g, ";\n")
        .replace(/\,(?![^\(|\"]*(\)|\"))/g, ",\n")
        .replace(/\n\s*\n/g, "\n")
        .split("\n");
    let depth = 0;
    const formatted = [];
    for (let x = 0, len = css.length; x < len; x++) {
        const line = `${css[x]}\n`;
        if (line.includes("{")) {
            formatted.push(getIndent(depth++) + trimLeft(line));
        } else if (line.includes("}")) {
            formatted.push(getIndent(--depth) + trimLeft(line));
        } else {
            formatted.push(getIndent(depth) + trimLeft(line));
```

```
        }
    }
    return formatted.join("").trim();
}
function getParenthesesRange(input, searchStartIndex = 0) {
    const length = input.length;
    let depth = 0;
    let firstOpenIndex = -1;
    for (let i = searchStartIndex; i < length; i++) {
        if (depth === 0) {
            const openIndex = input.indexOf("(", i);
            if (openIndex < 0) {
                break;
            }
            firstOpenIndex = openIndex;
            depth++;
            i = openIndex;
        } else {
            const closingIndex = input.indexOf(")", i);
            if (closingIndex < 0) {
                break;
            }
            const openIndex = input.indexOf("(", i);
            if (openIndex < 0 || closingIndex < openIndex) {
                depth--;
                if (depth === 0) {
                    return {start: firstOpenIndex, end: closingIndex + 1};
                }
                i = closingIndex;
            } else {
                depth++;
                i = openIndex;
            }
        }
    }
    return null;
}

const hslaParseCache = new Map();
const rgbaParseCache = new Map();
function parseColorWithCache($color) {
    $color = $color.trim();
    if (rgbaParseCache.has($color)) {
        return rgbaParseCache.get($color);
    }
    if ($color.includes("calc(")) {
        $color = lowerCalcExpression($color);
    }
    const color = parse($color);
    color && rgbaParseCache.set($color, color);
    return color;
}
function parseToHSLWithCache(color) {
    if (hslaParseCache.has(color)) {
        return hslaParseCache.get(color);
    }
    const rgb = parseColorWithCache(color);
    if (!rgb) {
        return null;
    }
    const hsl = rgbToHSL(rgb);
    hslaParseCache.set(color, hsl);
    return hsl;
}
function clearColorCache() {
    hslaParseCache.clear();
    rgbaParseCache.clear();
}
function hslToRGB({h, s, l, a = 1}) {
```

```
    if (s === 0) {
        const [r, b, g] = [l, l, l].map((x) => Math.round(x * 255));
        return {r, g, b, a};
    }
    const c = (1 - Math.abs(2 * l - 1)) * s;
    const x = c * (1 - Math.abs(((h / 60) % 2) - 1));
    const m = l - c / 2;
    const [r, g, b] = (
        h < 60
            ? [c, x, 0]
            : h < 120
              ? [x, c, 0]
              : h < 180
                ? [0, c, x]
                : h < 240
                  ? [0, x, c]
                  : h < 300
                    ? [x, 0, c]
                    : [c, 0, x]
    ).map((n) => Math.round((n + m) * 255));
    return {r, g, b, a};
}
function rgbToHSL({r: r255, g: g255, b: b255, a = 1}) {
    const r = r255 / 255;
    const g = g255 / 255;
    const b = b255 / 255;
    const max = Math.max(r, g, b);
    const min = Math.min(r, g, b);
    const c = max - min;
    const l = (max + min) / 2;
    if (c === 0) {
        return {h: 0, s: 0, l, a};
    }
    let h =
        (max === r
            ? ((g - b) / c) % 6
            : max === g
              ? (b - r) / c + 2
              : (r - g) / c + 4) * 60;
    if (h < 0) {
        h += 360;
    }
    const s = c / (1 - Math.abs(2 * l - 1));
    return {h, s, l, a};
}
function toFixed(n, digits = 0) {
    const fixed = n.toFixed(digits);
    if (digits === 0) {
        return fixed;
    }
    const dot = fixed.indexOf(".");
    if (dot >= 0) {
        const zerosMatch = fixed.match(/0+$/);
        if (zerosMatch) {
            if (zerosMatch.index === dot + 1) {
                return fixed.substring(0, dot);
            }
            return fixed.substring(0, zerosMatch.index);
        }
    }
    return fixed;
}
function rgbToString(rgb) {
    const {r, g, b, a} = rgb;
    if (a != null && a < 1) {
        return `rgba(${toFixed(r)}, ${toFixed(g)}, ${toFixed(b)}, ${toFixed(
            a,
            2
        )})`;
```

```
    }
    return `rgb(${toFixed(r)}, ${toFixed(g)}, ${toFixed(b)})`;
}
function rgbToHexString({r, g, b, a}) {
    return `#${(a != null && a < 1
        ? [r, g, b, Math.round(a * 255)]
        : [r, g, b]
    )
        .map((x) => {
            return `${x < 16 ? "0" : ""}${x.toString(16)}`;
        })
        .join("")}`;
}
function hslToString(hsl) {
    const {h, s, l, a} = hsl;
    if (a != null && a < 1) {
        return `hsla(${toFixed(h)}, ${toFixed(s * 100)}%, ${toFixed(
            l * 100
        )}%, ${toFixed(a, 2)})`;
    }
    return `hsl(${toFixed(h)}, ${toFixed(s * 100)}%, ${toFixed(l * 100)}%)`;
}
const rgbMatch = /^rgba?\([^\(\)]+\)$/;
const hslMatch = /^hsla?\([^\(\)]+\)$/;
const hexMatch = /^#[0-9a-f]+$/i;
function parse($color) {
    const c = $color.trim().toLowerCase();
    if (c.match(rgbMatch)) {
        return parseRGB(c);
    }
    if (c.match(hslMatch)) {
        return parseHSL(c);
    }
    if (c.match(hexMatch)) {
        return parseHex(c);
    }
    if (knownColors.has(c)) {
        return getColorByName(c);
    }
    if (systemColors.has(c)) {
        return getSystemColor(c);
    }
    if ($color === "transparent") {
        return {r: 0, g: 0, b: 0, a: 0};
    }
    return null;
}
function getNumbers($color) {
    const numbers = [];
    let prevPos = 0;
    let isMining = false;
    const startIndex = $color.indexOf("(");
    $color = $color.substring(startIndex + 1, $color.length - 1);
    for (let i = 0; i < $color.length; i++) {
        const c = $color[i];
        if ((c >= "0" && c <= "9") || c === "." || c === "+" || c === "-") {
            isMining = true;
        } else if (isMining && (c === " " || c === "," || c === "/")) {
            numbers.push($color.substring(prevPos, i));
            isMining = false;
            prevPos = i + 1;
        } else if (!isMining) {
            prevPos = i + 1;
        }
    }
    if (isMining) {
        numbers.push($color.substring(prevPos, $color.length));
    }
    return numbers;
```

```
}
function getNumbersFromString(str, range, units) {
    const raw = getNumbers(str);
    const unitsList = Object.entries(units);
    const numbers = raw
        .map((r) => r.trim())
        .map((r, i) => {
            let n;
            const unit = unitsList.find(([u]) => r.endsWith(u));
            if (unit) {
                n =
                    (parseFloat(r.substring(0, r.length - unit[0].length)) /
                        unit[1]) *
                    range[i];
            } else {
                n = parseFloat(r);
            }
            if (range[i] > 1) {
                return Math.round(n);
            }
            return n;
        });
    return numbers;
}
const rgbRange = [255, 255, 255, 1];
const rgbUnits = {"%": 100};
function parseRGB($rgb) {
    const [r, g, b, a = 1] = getNumbersFromString($rgb, rgbRange, rgbUnits);
    return {r, g, b, a};
}
const hslRange = [360, 1, 1, 1];
const hslUnits = {"%": 100, "deg": 360, "rad": 2 * Math.PI, "turn": 1};
function parseHSL($hsl) {
    const [h, s, l, a = 1] = getNumbersFromString($hsl, hslRange, hslUnits);
    return hslToRGB({h, s, l, a});
}
function parseHex($hex) {
    const h = $hex.substring(1);
    switch (h.length) {
        case 3:
        case 4: {
            const [r, g, b] = [0, 1, 2].map((i) =>
                parseInt(`${h[i]}${h[i]}`, 16)
            );
            const a =
                h.length === 3 ? 1 : parseInt(`${h[3]}${h[3]}`, 16) / 255;
            return {r, g, b, a};
        }
        case 6:
        case 8: {
            const [r, g, b] = [0, 2, 4].map((i) =>
                parseInt(h.substring(i, i + 2), 16)
            );
            const a =
                h.length === 6 ? 1 : parseInt(h.substring(6, 8), 16) / 255;
            return {r, g, b, a};
        }
    }
    return null;
}
function getColorByName($color) {
    const n = knownColors.get($color);
    return {
        r: (n >> 16) & 255,
        g: (n >> 8) & 255,
        b: (n >> 0) & 255,
        a: 1
    };
}
```

```
function getSystemColor($color) {
    const n = systemColors.get($color);
    return {
        r: (n >> 16) & 255,
        g: (n >> 8) & 255,
        b: (n >> 0) & 255,
        a: 1
    };
}
function lowerCalcExpression(color) {
    let searchIndex = 0;
    const replaceBetweenIndices = (start, end, replacement) => {
        color =
            color.substring(0, start) + replacement + color.substring(end);
    };
    while ((searchIndex = color.indexOf("calc(")) !== -1) {
        const range = getParenthesesRange(color, searchIndex);
        if (!range) {
            break;
        }
        let slice = color.slice(range.start + 1, range.end - 1);
        const includesPercentage = slice.includes("%");
        slice = slice.split("%").join("");
        const output = Math.round(evalMath(slice));
        replaceBetweenIndices(
            range.start - 4,
            range.end,
            output + (includesPercentage ? "%" : "")
        );
    }
    return color;
}
const knownColors = new Map(
    Object.entries({
        aliceblue: 0xf0f8ff,
        antiquewhite: 0xfaebd7,
        aqua: 0x00ffff,
        aquamarine: 0x7fffd4,
        azure: 0xf0ffff,
        beige: 0xf5f5dc,
        bisque: 0xffe4c4,
        black: 0x000000,
        blanchedalmond: 0xffebcd,
        blue: 0x0000ff,
        blueviolet: 0x8a2be2,
        brown: 0xa52a2a,
        burlywood: 0xdeb887,
        cadetblue: 0x5f9ea0,
        chartreuse: 0x7fff00,
        chocolate: 0xd2691e,
        coral: 0xff7f50,
        cornflowerblue: 0x6495ed,
        cornsilk: 0xfff8dc,
        crimson: 0xdc143c,
        cyan: 0x00ffff,
        darkblue: 0x00008b,
        darkcyan: 0x008b8b,
        darkgoldenrod: 0xb8860b,
        darkgray: 0xa9a9a9,
        darkgrey: 0xa9a9a9,
        darkgreen: 0x006400,
        darkkhaki: 0xbdb76b,
        darkmagenta: 0x8b008b,
        darkolivegreen: 0x556b2f,
        darkorange: 0xff8c00,
        darkorchid: 0x9932cc,
        darkred: 0x8b0000,
        darksalmon: 0xe9967a,
        darkseagreen: 0x8fbc8f,
```

```
darkslateblue: 0x483d8b,
darkslategray: 0x2f4f4f,
darkslategrey: 0x2f4f4f,
darkturquoise: 0x00ced1,
darkviolet: 0x9400d3,
deeppink: 0xff1493,
deepskyblue: 0x00bfff,
dimgray: 0x696969,
dimgrey: 0x696969,
dodgerblue: 0x1e90ff,
firebrick: 0xb22222,
floralwhite: 0xfffaf0,
forestgreen: 0x228b22,
fuchsia: 0xff00ff,
gainsboro: 0xdcdcdc,
ghostwhite: 0xf8f8ff,
gold: 0xffd700,
goldenrod: 0xdaa520,
gray: 0x808080,
grey: 0x808080,
green: 0x008000,
greenyellow: 0xadff2f,
honeydew: 0xf0fff0,
hotpink: 0xff69b4,
indianred: 0xcd5c5c,
indigo: 0x4b0082,
ivory: 0xfffff0,
khaki: 0xf0e68c,
lavender: 0xe6e6fa,
lavenderblush: 0xfff0f5,
lawngreen: 0x7cfc00,
lemonchiffon: 0xfffacd,
lightblue: 0xadd8e6,
lightcoral: 0xf08080,
lightcyan: 0xe0ffff,
lightgoldenrodyellow: 0xfafad2,
lightgray: 0xd3d3d3,
lightgrey: 0xd3d3d3,
lightgreen: 0x90ee90,
lightpink: 0xffb6c1,
lightsalmon: 0xffa07a,
lightseagreen: 0x20b2aa,
lightskyblue: 0x87cefa,
lightslategray: 0x778899,
lightslategrey: 0x778899,
lightsteelblue: 0xb0c4de,
lightyellow: 0xffffe0,
lime: 0x00ff00,
limegreen: 0x32cd32,
linen: 0xfaf0e6,
magenta: 0xff00ff,
maroon: 0x800000,
mediumaquamarine: 0x66cdaa,
mediumblue: 0x0000cd,
mediumorchid: 0xba55d3,
mediumpurple: 0x9370db,
mediumseagreen: 0x3cb371,
mediumslateblue: 0x7b68ee,
mediumspringgreen: 0x00fa9a,
mediumturquoise: 0x48d1cc,
mediumvioletred: 0xc71585,
midnightblue: 0x191970,
mintcream: 0xf5fffa,
mistyrose: 0xffe4e1,
moccasin: 0xffe4b5,
navajowhite: 0xffdead,
navy: 0x000080,
oldlace: 0xfdf5e6,
olive: 0x808000,
```

```
        olivedrab: 0x6b8e23,
        orange: 0xffa500,
        orangered: 0xff4500,
        orchid: 0xda70d6,
        palegoldenrod: 0xeee8aa,
        palegreen: 0x98fb98,
        paleturquoise: 0xafeeee,
        palevioletred: 0xdb7093,
        papayawhip: 0xffefd5,
        peachpuff: 0xffdab9,
        peru: 0xcd853f,
        pink: 0xffc0cb,
        plum: 0xdda0dd,
        powderblue: 0xb0e0e6,
        purple: 0x800080,
        rebeccapurple: 0x663399,
        red: 0xff0000,
        rosybrown: 0xbc8f8f,
        royalblue: 0x4169e1,
        saddlebrown: 0x8b4513,
        salmon: 0xfa8072,
        sandybrown: 0xf4a460,
        seagreen: 0x2e8b57,
        seashell: 0xfff5ee,
        sienna: 0xa0522d,
        silver: 0xc0c0c0,
        skyblue: 0x87ceeb,
        slateblue: 0x6a5acd,
        slategray: 0x708090,
        slategrey: 0x708090,
        snow: 0xfffafa,
        springgreen: 0x00ff7f,
        steelblue: 0x4682b4,
        tan: 0xd2b48c,
        teal: 0x008080,
        thistle: 0xd8bfd8,
        tomato: 0xff6347,
        turquoise: 0x40e0d0,
        violet: 0xee82ee,
        wheat: 0xf5deb3,
        white: 0xffffff,
        whitesmoke: 0xf5f5f5,
        yellow: 0xffff00,
        yellowgreen: 0x9acd32
    })
);
const systemColors = new Map(
    Object.entries({
        "ActiveBorder": 0x3b99fc,
        "ActiveCaption": 0x000000,
        "AppWorkspace": 0xaaaaaa,
        "Background": 0x6363ce,
        "ButtonFace": 0xffffff,
        "ButtonHighlight": 0xe9e9e9,
        "ButtonShadow": 0x9fa09f,
        "ButtonText": 0x000000,
        "CaptionText": 0x000000,
        "GrayText": 0x7f7f7f,
        "Highlight": 0xb2d7ff,
        "HighlightText": 0x000000,
        "InactiveBorder": 0xffffff,
        "InactiveCaption": 0xffffff,
        "InactiveCaptionText": 0x000000,
        "InfoBackground": 0xfbfcc5,
        "InfoText": 0x000000,
        "Menu": 0xf6f6f6,
        "MenuText": 0xffffff,
        "Scrollbar": 0xaaaaaa,
        "ThreeDDarkShadow": 0x000000,
```

```
        "ThreeDFace": 0xc0c0c0,
        "ThreeDHighlight": 0xffffff,
        "ThreeDLightShadow": 0xffffff,
        "ThreeDShadow": 0x000000,
        "Window": 0xecececec,
        "WindowFrame": 0xaaaaaa,
        "WindowText": 0x000000,
        "-webkit-focus-ring-color": 0xe59700
    }).map(([key, value]) => [key.toLowerCase(), value])
);
function getSRGBLightness(r, g, b) {
    return (0.2126 * r + 0.7152 * g + 0.0722 * b) / 255;
}

const isNavigatorDefined = typeof navigator !== "undefined";
const userAgent = isNavigatorDefined
    ? navigator.userAgentData &&
      Array.isArray(navigator.userAgentData.brands)
        ? navigator.userAgentData.brands
              .map(
                  (brand) => `${brand.brand.toLowerCase()} ${brand.version}`
              )
              .join(" ")
        : navigator.userAgent.toLowerCase()
    : "some useragent";
const platform = isNavigatorDefined
    ? navigator.userAgentData &&
      typeof navigator.userAgentData.platform === "string"
        ? navigator.userAgentData.platform.toLowerCase()
        : navigator.platform.toLowerCase()
    : "some platform";
userAgent.includes("vivaldi");
userAgent.includes("yabrowser");
userAgent.includes("opr") || userAgent.includes("opera");
userAgent.includes("edg");
platform.startsWith("win");
platform.startsWith("mac");
isNavigatorDefined && navigator.userAgentData
    ? navigator.userAgentData.mobile
    : userAgent.includes("mobile");
const isShadowDomSupported = typeof ShadowRoot === "function";
const isMatchMediaChangeEventListenerSupported =
    typeof MediaQueryList === "function" &&
    typeof MediaQueryList.prototype.addEventListener === "function";
(isNavigatorDefined &&
    navigator.userAgentData &&
    ["Linux", "Android"].includes(navigator.userAgentData.platform)) ||
    platform.startsWith("linux");
(() => {
    const m = userAgent.match(/chrom(?:e|ium)(?:\/| )([^ ]+)/);
    if (m && m[1]) {
        return m[1];
    }
    return "";
})();
(() => {
    const m = userAgent.match(/(?:firefox|librewolf)(?:\/| )([^ ]+)/);
    if (m && m[1]) {
        return m[1];
    }
    return "";
})();
const isDefinedSelectorSupported = (() => {
    try {
        document.querySelector(":defined");
        return true;
    } catch (err) {
        return false;
    }
```

```
})();
const isCSSColorSchemePropSupported = (() => {
    try {
        if (typeof document === "undefined") {
            return false;
        }
        const el = document.createElement("div");
        if (!el || typeof el.style !== "object") {
            return false;
        }
        if (typeof el.style.colorScheme === "string") {
            return true;
        }
        el.setAttribute("style", "color-scheme: dark");
        return el.style.colorScheme === "dark";
    } catch (e) {
        return false;
    }
})();

let query = null;
const onChange = ({matches}) =>
    listeners.forEach((listener) => listener(matches));
const listeners = new Set();
function runColorSchemeChangeDetector(callback) {
    listeners.add(callback);
    if (query) {
        return;
    }
    query = matchMedia("(prefers-color-scheme: dark)");
    if (isMatchMediaChangeEventListenerSupported) {
        query.addEventListener("change", onChange);
    } else {
        query.addListener(onChange);
    }
}
function stopColorSchemeChangeDetector() {
    if (!query || !onChange) {
        return;
    }
    if (isMatchMediaChangeEventListenerSupported) {
        query.removeEventListener("change", onChange);
    } else {
        query.removeListener(onChange);
    }
    listeners.clear();
    query = null;
}
const isSystemDarkModeEnabled = () =>
    (query || matchMedia("(prefers-color-scheme: dark)")).matches;

const COLOR_SCHEME_META_SELECTOR = 'meta[name="color-scheme"]';
function hasBuiltInDarkTheme() {
    const rootStyle = getComputedStyle(document.documentElement);
    if (rootStyle.filter.includes("invert(1)")) {
        return true;
    }
    const CELL_SIZE = 256;
    const MAX_ROW_COUNT = 4;
    const winWidth = innerWidth;
    const winHeight = innerHeight;
    const stepX = Math.floor(
        winWidth / Math.min(MAX_ROW_COUNT, Math.ceil(winWidth / CELL_SIZE))
    );
    const stepY = Math.floor(
        winHeight /
            Math.min(MAX_ROW_COUNT, Math.ceil(winHeight / CELL_SIZE))
    );
    const processedElements = new Set();
```

```
    for (let y = Math.floor(stepY / 2); y < winHeight; y += stepY) {
        for (let x = Math.floor(stepX / 2); x < winWidth; x += stepX) {
            const element = document.elementFromPoint(x, y);
            if (!element || processedElements.has(element)) {
                continue;
            }
            processedElements.add(element);
            const style =
                element === document.documentElement
                    ? rootStyle
                    : getComputedStyle(element);
            const bgColor = parseColorWithCache(style.backgroundColor);
            if (bgColor.a === 1) {
                const bgLightness = getSRGBLightness(
                    bgColor.r,
                    bgColor.g,
                    bgColor.b
                );
                if (bgLightness > 0.5) {
                    return false;
                }
            } else {
                const textColor = parseColorWithCache(style.color);
                const textLightness = getSRGBLightness(
                    textColor.r,
                    textColor.g,
                    textColor.b
                );
                if (textLightness < 0.5) {
                    return false;
                }
            }
        }
    }
    const rootColor = parseColorWithCache(rootStyle.backgroundColor);
    const bodyColor = document.body
        ? parseColorWithCache(
              getComputedStyle(document.body).backgroundColor
          )
        : {r: 0, g: 0, b: 0, a: 0};
    const rootLightness =
        1 -
        rootColor.a +
        rootColor.a *
            getSRGBLightness(rootColor.r, rootColor.g, rootColor.b);
    const finalLightness =
        (1 - bodyColor.a) * rootLightness +
        bodyColor.a *
            getSRGBLightness(bodyColor.r, bodyColor.g, bodyColor.b);
    return finalLightness < 0.5;
}
function runCheck(callback) {
    const colorSchemeMeta = document.querySelector(
        COLOR_SCHEME_META_SELECTOR
    );
    if (colorSchemeMeta) {
        const isMetaDark =
            colorSchemeMeta.content === "dark" ||
            (colorSchemeMeta.content.includes("dark") &&
                isSystemDarkModeEnabled());
        callback(isMetaDark);
        return;
    }
    const drStyles = document.querySelectorAll(".darkreader");
    drStyles.forEach((style) => (style.disabled = true));
    const darkThemeDetected = hasBuiltInDarkTheme();
    drStyles.forEach((style) => (style.disabled = false));
    callback(darkThemeDetected);
}
```

```
function hasSomeStyle() {
    if (document.querySelector(COLOR_SCHEME_META_SELECTOR) != null) {
        return true;
    }
    if (
        document.documentElement.style.backgroundColor ||
        (document.body && document.body.style.backgroundColor)
    ) {
        return true;
    }
    for (const style of document.styleSheets) {
        if (
            style &&
            style.ownerNode &&
            !(
                style.ownerNode.classList &&
                style.ownerNode.classList.contains("darkreader")
            )
        ) {
            return true;
        }
    }
    return false;
}
let observer$1;
let readyStateListener;
function runDarkThemeDetector(callback, hints) {
    stopDarkThemeDetector();
    if (hints && hints.length > 0) {
        const hint = hints[0];
        if (hint.noDarkTheme) {
            callback(false);
            return;
        }
        if (hint.systemTheme && isSystemDarkModeEnabled()) {
            callback(true);
            return;
        }
        detectUsingHint(hint, () => callback(true));
        return;
    }
    if (document.body && hasSomeStyle()) {
        runCheck(callback);
        return;
    }
    observer$1 = new MutationObserver(() => {
        if (document.body && hasSomeStyle()) {
            stopDarkThemeDetector();
            runCheck(callback);
        }
    });
    observer$1.observe(document.documentElement, {childList: true});
    if (document.readyState !== "complete") {
        readyStateListener = () => {
            if (document.readyState === "complete") {
                stopDarkThemeDetector();
                runCheck(callback);
            }
        };
        document.addEventListener("readystatechange", readyStateListener);
    }
}
function stopDarkThemeDetector() {
    if (observer$1) {
        observer$1.disconnect();
        observer$1 = null;
    }
    if (readyStateListener) {
        document.removeEventListener(
```

```
            "readystatechange",
            readyStateListener
        );
        readyStateListener = null;
    }
    stopDetectingUsingHint();
}
let hintTargetObserver;
let hintMatchObserver;
function detectUsingHint(hint, success) {
    stopDetectingUsingHint();
    const matchSelector = (hint.match || []).join(", ");
    function checkMatch(target) {
        var _a;
        if (
            (_a = target.matches) === null || _a === void 0
                ? void 0
                : _a.call(target, matchSelector)
        ) {
            stopDetectingUsingHint();
            success();
            return true;
        }
        return false;
    }
    function setupMatchObserver(target) {
        hintMatchObserver === null || hintMatchObserver === void 0
            ? void 0
            : hintMatchObserver.disconnect();
        if (checkMatch(target)) {
            return;
        }
        hintMatchObserver = new MutationObserver(() => checkMatch(target));
        hintMatchObserver.observe(target, {attributes: true});
    }
    const target = document.querySelector(hint.target);
    if (target) {
        setupMatchObserver(target);
    } else {
        hintTargetObserver = new MutationObserver((mutations) => {
            const handledTargets = new Set();
            for (const mutation of mutations) {
                if (handledTargets.has(mutation.target)) {
                    continue;
                }
                handledTargets.add(mutation.target);
                if (mutation.target instanceof Element) {
                    const target = mutation.target.querySelector(
                        hint.target
                    );
                    if (target) {
                        hintTargetObserver.disconnect();
                        setupMatchObserver(target);
                        break;
                    }
                }
            }
        });
        hintTargetObserver.observe(document.documentElement, {
            childList: true,
            subtree: true
        });
    }
}
function stopDetectingUsingHint() {
    hintTargetObserver === null || hintTargetObserver === void 0
        ? void 0
        : hintTargetObserver.disconnect();
    hintMatchObserver === null || hintMatchObserver === void 0
```

```
        ? void 0
        : hintMatchObserver.disconnect();
}

function cachedFactory(factory, size) {
    const cache = new Map();
    return (key) => {
        if (cache.has(key)) {
            return cache.get(key);
        }
        const value = factory(key);
        cache.set(key, value);
        if (cache.size > size) {
            const first = cache.keys().next().value;
            cache.delete(first);
        }
        return value;
    };
}

const simpleIPV6Regex = /\[[0-9:a-zA-Z]+?\]/;
function isIPV6(url) {
    const openingBracketIndex = simpleIPV6Regex.exec(url);
    if (!openingBracketIndex) {
        return false;
    }
    const queryIndex = url.indexOf("?");
    if (queryIndex >= 0 && openingBracketIndex.index > queryIndex) {
        return false;
    }
    return true;
}
const ipV6HostRegex = /\[.*?\](\:\d+)?/;
function compareIPV6(firstURL, secondURL) {
    const firstHost = firstURL.match(ipV6HostRegex)[0];
    const secondHost = secondURL.match(ipV6HostRegex)[0];
    return firstHost === secondHost;
}

let anchor;
const parsedURLCache = new Map();
function fixBaseURL($url) {
    if (!anchor) {
        anchor = document.createElement("a");
    }
    anchor.href = $url;
    return anchor.href;
}
function parseURL($url, $base = null) {
    const key = `${$url}${$base ? `;${$base}` : ""}`;
    if (parsedURLCache.has(key)) {
        return parsedURLCache.get(key);
    }
    if ($base) {
        const parsedURL = new URL($url, fixBaseURL($base));
        parsedURLCache.set(key, parsedURL);
        return parsedURL;
    }
    const parsedURL = new URL(fixBaseURL($url));
    parsedURLCache.set($url, parsedURL);
    return parsedURL;
}
function getAbsoluteURL($base, $relative) {
    if ($relative.match(/^data\\?\:/)) {
        return $relative;
    }
    if (/^\/\//.test($relative)) {
        return `${location.protocol}${$relative}`;
    }
```

```
    const b = parseURL($base);
    const a = parseURL($relative, b.href);
    return a.href;
}
function isRelativeHrefOnAbsolutePath(href) {
    if (href.startsWith("data:")) {
        return true;
    }
    const url = parseURL(href);
    if (url.protocol !== location.protocol) {
        return false;
    }
    if (url.hostname !== location.hostname) {
        return false;
    }
    if (url.port !== location.port) {
        return false;
    }
    return url.pathname === location.pathname;
}
function isURLInList(url, list) {
    for (let i = 0; i < list.length; i++) {
        if (isURLMatched(url, list[i])) {
            return true;
        }
    }
    return false;
}
function isURLMatched(url, urlTemplate) {
    const isFirstIPV6 = isIPV6(url);
    const isSecondIPV6 = isIPV6(urlTemplate);
    if (isRegExp(urlTemplate)) {
        const regexp = createRegExp(urlTemplate);
        return regexp ? regexp.test(url) : false;
    } else if (isFirstIPV6 && isSecondIPV6) {
        return compareIPV6(url, urlTemplate);
    } else if (!isFirstIPV6 && !isSecondIPV6) {
        return matchURLPattern(url, urlTemplate);
    }
    return false;
}
const URL_CACHE_SIZE = 32;
const prepareURL = cachedFactory((url) => {
    let parsed;
    try {
        parsed = new URL(url);
    } catch (err) {
        return null;
    }
    const {hostname, pathname, protocol, port} = parsed;
    const hostParts = hostname.split(".").reverse();
    const pathParts = pathname.split("/").slice(1);
    if (!pathParts[pathParts.length - 1]) {
        pathParts.splice(pathParts.length - 1, 1);
    }
    return {
        hostParts,
        pathParts,
        port,
        protocol
    };
}, URL_CACHE_SIZE);
const URL_MATCH_CACHE_SIZE = 32 * 1024;
const preparePattern = cachedFactory((pattern) => {
    if (!pattern) {
        return null;
    }
    const exactStart = pattern.startsWith("^");
    const exactEnd = pattern.endsWith("$");
```

```
    if (exactStart) {
        pattern = pattern.substring(1);
    }
    if (exactEnd) {
        pattern = pattern.substring(0, pattern.length - 1);
    }
    let protocol = "";
    const protocolIndex = pattern.indexOf("://");
    if (protocolIndex > 0) {
        protocol = pattern.substring(0, protocolIndex + 1);
        pattern = pattern.substring(protocolIndex + 3);
    }
    const slashIndex = pattern.indexOf("/");
    const host =
        slashIndex < 0 ? pattern : pattern.substring(0, slashIndex);
    let hostName = host;
    let port = "*";
    const portIndex = host.indexOf(":");
    if (portIndex >= 0) {
        hostName = host.substring(0, portIndex);
        port = host.substring(portIndex + 1);
    }
    const hostParts = hostName.split(".").reverse();
    const path = slashIndex < 0 ? "" : pattern.substring(slashIndex + 1);
    const pathParts = path.split("/");
    if (!pathParts[pathParts.length - 1]) {
        pathParts.splice(pathParts.length - 1, 1);
    }
    return {
        hostParts,
        pathParts,
        port,
        exactStart,
        exactEnd,
        protocol
    };
}, URL_MATCH_CACHE_SIZE);
function matchURLPattern(url, pattern) {
    const u = prepareURL(url);
    const p = preparePattern(pattern);
    if (
        !(u && p) ||
        p.hostParts.length > u.hostParts.length ||
        (p.exactStart && p.hostParts.length !== u.hostParts.length) ||
        (p.exactEnd && p.pathParts.length !== u.pathParts.length) ||
        (p.port !== "*" && p.port !== u.port) ||
        (p.protocol && p.protocol !== u.protocol)
    ) {
        return false;
    }
    for (let i = 0; i < p.hostParts.length; i++) {
        const pHostPart = p.hostParts[i];
        const uHostPart = u.hostParts[i];
        if (pHostPart !== "*" && pHostPart !== uHostPart) {
            return false;
        }
    }
    if (
        p.hostParts.length >= 2 &&
        p.hostParts.at(-1) !== "*" &&
        (p.hostParts.length < u.hostParts.length - 1 ||
            (p.hostParts.length === u.hostParts.length - 1 &&
                u.hostParts.at(-1) !== "www"))
    ) {
        return false;
    }
    if (p.pathParts.length === 0) {
        return true;
    }
```

```
    if (p.pathParts.length > u.pathParts.length) {
        return false;
    }
    for (let i = 0; i < p.pathParts.length; i++) {
        const pPathPart = p.pathParts[i];
        const uPathPart = u.pathParts[i];
        if (pPathPart !== "*" && pPathPart !== uPathPart) {
            return false;
        }
    }
    return true;
}
function isRegExp(pattern) {
    return (
        pattern.startsWith("/") &&
        pattern.endsWith("/") &&
        pattern.length > 2
    );
}
const REGEXP_CACHE_SIZE = 1024;
const createRegExp = cachedFactory((pattern) => {
    if (pattern.startsWith("/")) {
        pattern = pattern.substring(1);
    }
    if (pattern.endsWith("/")) {
        pattern = pattern.substring(0, pattern.length - 1);
    }
    try {
        return new RegExp(pattern);
    } catch (err) {
        return null;
    }
}, REGEXP_CACHE_SIZE);

function iterateCSSRules(rules, iterate, onMediaRuleError) {
    forEach(rules, (rule) => {
        if (rule.selectorText) {
            iterate(rule);
        } else if (rule.href) {
            try {
                iterateCSSRules(
                    rule.styleSheet.cssRules,
                    iterate,
                    onMediaRuleError
                );
            } catch (err) {
                onMediaRuleError && onMediaRuleError();
            }
        } else if (rule.media) {
            const media = Array.from(rule.media);
            const isScreenOrAllOrQuery = media.some(
                (m) =>
                    m.startsWith("screen") ||
                    m.startsWith("all") ||
                    m.startsWith("(")
            );
            const isPrintOrSpeech = media.some(
                (m) => m.startsWith("print") || m.startsWith("speech")
            );
            if (isScreenOrAllOrQuery || !isPrintOrSpeech) {
                iterateCSSRules(rule.cssRules, iterate, onMediaRuleError);
            }
        } else if (rule.conditionText) {
            if (CSS.supports(rule.conditionText)) {
                iterateCSSRules(rule.cssRules, iterate, onMediaRuleError);
            }
        } else;
    });
}
```

```
const shorthandVarDependantProperties = [
    "background",
    "border",
    "border-color",
    "border-bottom",
    "border-left",
    "border-right",
    "border-top",
    "outline",
    "outline-color"
];
function iterateCSSDeclarations(style, iterate) {
    forEach(style, (property) => {
        const value = style.getPropertyValue(property).trim();
        if (!value) {
            return;
        }
        iterate(property, value);
    });
    const cssText = style.cssText;
    if (cssText.includes("var(")) {
        {
            shorthandVarDependantProperties.forEach((prop) => {
                const val = style.getPropertyValue(prop);
                if (val && val.includes("var(")) {
                    iterate(prop, val);
                }
            });
        }
    }
}
const cssURLRegex = /url\((('.*?')|(".*?")|([^\)]*?))\)/g;
const cssImportRegex =
    /@import\s*(url\()?(('.+?')|(".+?")|([^\)]*?))\)? ?(screen)?;?/gi;
function getCSSURLValue(cssURL) {
    return cssURL
        .trim()
        .replace(/[\n\r\\]+/g, "")
        .replace(/^url\((.*)\)$/, "$1")
        .trim()
        .replace(/^"(.*)"$/, "$1")
        .replace(/^'(.*)'$/, "$1")
        .replace(/(?:\\(.))/g, "$1");
}
function getCSSBaseBath(url) {
    const cssURL = parseURL(url);
    return `${cssURL.origin}${cssURL.pathname
        .replace(/\?.*$/, "")
        .replace(/(\/)([^\/]+)$/i, "$1")}`;
}
function replaceCSSRelativeURLsWithAbsolute($css, cssBasePath) {
    return $css.replace(cssURLRegex, (match) => {
        const pathValue = getCSSURLValue(match);
        try {
            return `url('${getAbsoluteURL(cssBasePath, pathValue)}')`;
        } catch (err) {
            return match;
        }
    });
}
const cssCommentsRegex = /\/\*[\s\S]*?\*\//g;
function removeCSSComments($css) {
    return $css.replace(cssCommentsRegex, "");
}
const fontFaceRegex = /@font-face\s*{[^}]*}/g;
function replaceCSSFontFace($css) {
    return $css.replace(fontFaceRegex, "");
}
```

```
function scale(x, inLow, inHigh, outLow, outHigh) {
    return ((x - inLow) * (outHigh - outLow)) / (inHigh - inLow) + outLow;
}
function clamp(x, min, max) {
    return Math.min(max, Math.max(min, x));
}
function multiplyMatrices(m1, m2) {
    const result = [];
    for (let i = 0, len = m1.length; i < len; i++) {
        result[i] = [];
        for (let j = 0, len2 = m2[0].length; j < len2; j++) {
            let sum = 0;
            for (let k = 0, len3 = m1[0].length; k < len3; k++) {
                sum += m1[i][k] * m2[k][j];
            }
            result[i][j] = sum;
        }
    }
    return result;
}

function createFilterMatrix(config) {
    let m = Matrix.identity();
    if (config.sepia !== 0) {
        m = multiplyMatrices(m, Matrix.sepia(config.sepia / 100));
    }
    if (config.grayscale !== 0) {
        m = multiplyMatrices(m, Matrix.grayscale(config.grayscale / 100));
    }
    if (config.contrast !== 100) {
        m = multiplyMatrices(m, Matrix.contrast(config.contrast / 100));
    }
    if (config.brightness !== 100) {
        m = multiplyMatrices(m, Matrix.brightness(config.brightness / 100));
    }
    if (config.mode === 1) {
        m = multiplyMatrices(m, Matrix.invertNHue());
    }
    return m;
}
function applyColorMatrix([r, g, b], matrix) {
    const rgb = [[r / 255], [g / 255], [b / 255], [1], [1]];
    const result = multiplyMatrices(matrix, rgb);
    return [0, 1, 2].map((i) =>
        clamp(Math.round(result[i][0] * 255), 0, 255)
    );
}
const Matrix = {
    identity() {
        return [
            [1, 0, 0, 0, 0],
            [0, 1, 0, 0, 0],
            [0, 0, 1, 0, 0],
            [0, 0, 0, 1, 0],
            [0, 0, 0, 0, 1]
        ];
    },
    invertNHue() {
        return [
            [0.333, -0.667, -0.667, 0, 1],
            [-0.667, 0.333, -0.667, 0, 1],
            [-0.667, -0.667, 0.333, 0, 1],
            [0, 0, 0, 1, 0],
            [0, 0, 0, 0, 1]
        ];
    },
    brightness(v) {
        return [
            [v, 0, 0, 0, 0],
```

```
        [0, v, 0, 0, 0],
        [0, 0, v, 0, 0],
        [0, 0, 0, 1, 0],
        [0, 0, 0, 0, 1]
    ];
},
contrast(v) {
    const t = (1 - v) / 2;
    return [
        [v, 0, 0, 0, t],
        [0, v, 0, 0, t],
        [0, 0, v, 0, t],
        [0, 0, 0, 1, 0],
        [0, 0, 0, 0, 1]
    ];
},
sepia(v) {
    return [
        [
            0.393 + 0.607 * (1 - v),
            0.769 - 0.769 * (1 - v),
            0.189 - 0.189 * (1 - v),
            0,
            0
        ],
        [
            0.349 - 0.349 * (1 - v),
            0.686 + 0.314 * (1 - v),
            0.168 - 0.168 * (1 - v),
            0,
            0
        ],
        [
            0.272 - 0.272 * (1 - v),
            0.534 - 0.534 * (1 - v),
            0.131 + 0.869 * (1 - v),
            0,
            0
        ],
        [0, 0, 0, 1, 0],
        [0, 0, 0, 0, 1]
    ];
},
grayscale(v) {
    return [
        [
            0.2126 + 0.7874 * (1 - v),
            0.7152 - 0.7152 * (1 - v),
            0.0722 - 0.0722 * (1 - v),
            0,
            0
        ],
        [
            0.2126 - 0.2126 * (1 - v),
            0.7152 + 0.2848 * (1 - v),
            0.0722 - 0.0722 * (1 - v),
            0,
            0
        ],
        [
            0.2126 - 0.2126 * (1 - v),
            0.7152 - 0.7152 * (1 - v),
            0.0722 + 0.9278 * (1 - v),
            0,
            0
        ],
        [0, 0, 0, 1, 0],
        [0, 0, 0, 0, 1]
    ];
```

```
    }
};

function getBgPole(theme) {
    const isDarkScheme = theme.mode === 1;
    const prop = isDarkScheme
        ? "darkSchemeBackgroundColor"
        : "lightSchemeBackgroundColor";
    return theme[prop];
}
function getFgPole(theme) {
    const isDarkScheme = theme.mode === 1;
    const prop = isDarkScheme
        ? "darkSchemeTextColor"
        : "lightSchemeTextColor";
    return theme[prop];
}
const colorModificationCache = new Map();
function clearColorModificationCache() {
    colorModificationCache.clear();
}
const rgbCacheKeys = ["r", "g", "b", "a"];
const themeCacheKeys$1 = [
    "mode",
    "brightness",
    "contrast",
    "grayscale",
    "sepia",
    "darkSchemeBackgroundColor",
    "darkSchemeTextColor",
    "lightSchemeBackgroundColor",
    "lightSchemeTextColor"
];
function getCacheId(rgb, theme) {
    let resultId = "";
    rgbCacheKeys.forEach((key) => {
        resultId += `${rgb[key]};`;
    });
    themeCacheKeys$1.forEach((key) => {
        resultId += `${theme[key]};`;
    });
    return resultId;
}
function modifyColorWithCache(
    rgb,
    theme,
    modifyHSL,
    poleColor,
    anotherPoleColor
) {
    let fnCache;
    if (colorModificationCache.has(modifyHSL)) {
        fnCache = colorModificationCache.get(modifyHSL);
    } else {
        fnCache = new Map();
        colorModificationCache.set(modifyHSL, fnCache);
    }
    const id = getCacheId(rgb, theme);
    if (fnCache.has(id)) {
        return fnCache.get(id);
    }
    const hsl = rgbToHSL(rgb);
    const pole = poleColor == null ? null : parseToHSLWithCache(poleColor);
    const anotherPole =
        anotherPoleColor == null
            ? null
            : parseToHSLWithCache(anotherPoleColor);
    const modified = modifyHSL(hsl, pole, anotherPole);
    const {r, g, b, a} = hslToRGB(modified);
```

```
    const matrix = createFilterMatrix(theme);
    const [rf, gf, bf] = applyColorMatrix([r, g, b], matrix);
    const color =
        a === 1
            ? rgbToHexString({r: rf, g: gf, b: bf})
            : rgbToString({r: rf, g: gf, b: bf, a});
    fnCache.set(id, color);
    return color;
}
function noopHSL(hsl) {
    return hsl;
}
function modifyColor(rgb, theme) {
    return modifyColorWithCache(rgb, theme, noopHSL);
}
function modifyLightSchemeColor(rgb, theme) {
    const poleBg = getBgPole(theme);
    const poleFg = getFgPole(theme);
    return modifyColorWithCache(
        rgb,
        theme,
        modifyLightModeHSL,
        poleFg,
        poleBg
    );
}
function modifyLightModeHSL({h, s, l, a}, poleFg, poleBg) {
    const isDark = l < 0.5;
    let isNeutral;
    if (isDark) {
        isNeutral = l < 0.2 || s < 0.12;
    } else {
        const isBlue = h > 200 && h < 280;
        isNeutral = s < 0.24 || (l > 0.8 && isBlue);
    }
    let hx = h;
    let sx = l;
    if (isNeutral) {
        if (isDark) {
            hx = poleFg.h;
            sx = poleFg.s;
        } else {
            hx = poleBg.h;
            sx = poleBg.s;
        }
    }
    const lx = scale(l, 0, 1, poleFg.l, poleBg.l);
    return {h: hx, s: sx, l: lx, a};
}
const MAX_BG_LIGHTNESS = 0.4;
function modifyBgHSL({h, s, l, a}, pole) {
    const isDark = l < 0.5;
    const isBlue = h > 200 && h < 280;
    const isNeutral = s < 0.12 || (l > 0.8 && isBlue);
    if (isDark) {
        const lx = scale(l, 0, 0.5, 0, MAX_BG_LIGHTNESS);
        if (isNeutral) {
            const hx = pole.h;
            const sx = pole.s;
            return {h: hx, s: sx, l: lx, a};
        }
        return {h, s, l: lx, a};
    }
    let lx = scale(l, 0.5, 1, MAX_BG_LIGHTNESS, pole.l);
    if (isNeutral) {
        const hx = pole.h;
        const sx = pole.s;
        return {h: hx, s: sx, l: lx, a};
    }
```

```
    let hx = h;
    const isYellow = h > 60 && h < 180;
    if (isYellow) {
        const isCloserToGreen = h > 120;
        if (isCloserToGreen) {
            hx = scale(h, 120, 180, 135, 180);
        } else {
            hx = scale(h, 60, 120, 60, 105);
        }
    }
    if (hx > 40 && hx < 80) {
        lx *= 0.75;
    }
    return {h: hx, s, l: lx, a};
}
function modifyBackgroundColor(rgb, theme) {
    if (theme.mode === 0) {
        return modifyLightSchemeColor(rgb, theme);
    }
    const pole = getBgPole(theme);
    return modifyColorWithCache(
        rgb,
        {...theme, mode: 0},
        modifyBgHSL,
        pole
    );
}
const MIN_FG_LIGHTNESS = 0.55;
function modifyBlueFgHue(hue) {
    return scale(hue, 205, 245, 205, 220);
}
function modifyFgHSL({h, s, l, a}, pole) {
    const isLight = l > 0.5;
    const isNeutral = l < 0.2 || s < 0.24;
    const isBlue = !isNeutral && h > 205 && h < 245;
    if (isLight) {
        const lx = scale(l, 0.5, 1, MIN_FG_LIGHTNESS, pole.l);
        if (isNeutral) {
            const hx = pole.h;
            const sx = pole.s;
            return {h: hx, s: sx, l: lx, a};
        }
        let hx = h;
        if (isBlue) {
            hx = modifyBlueFgHue(h);
        }
        return {h: hx, s, l: lx, a};
    }
    if (isNeutral) {
        const hx = pole.h;
        const sx = pole.s;
        const lx = scale(l, 0, 0.5, pole.l, MIN_FG_LIGHTNESS);
        return {h: hx, s: sx, l: lx, a};
    }
    let hx = h;
    let lx;
    if (isBlue) {
        hx = modifyBlueFgHue(h);
        lx = scale(l, 0, 0.5, pole.l, Math.min(1, MIN_FG_LIGHTNESS + 0.05));
    } else {
        lx = scale(l, 0, 0.5, pole.l, MIN_FG_LIGHTNESS);
    }
    return {h: hx, s, l: lx, a};
}
function modifyForegroundColor(rgb, theme) {
    if (theme.mode === 0) {
        return modifyLightSchemeColor(rgb, theme);
    }
    const pole = getFgPole(theme);
```

```
        return modifyColorWithCache(
            rgb,
            {...theme, mode: 0},
            modifyFgHSL,
            pole
        );
    }
    function modifyBorderHSL({h, s, l, a}, poleFg, poleBg) {
        const isDark = l < 0.5;
        const isNeutral = l < 0.2 || s < 0.24;
        let hx = h;
        let sx = s;
        if (isNeutral) {
            if (isDark) {
                hx = poleFg.h;
                sx = poleFg.s;
            } else {
                hx = poleBg.h;
                sx = poleBg.s;
            }
        }
        const lx = scale(l, 0, 1, 0.5, 0.2);
        return {h: hx, s: sx, l: lx, a};
    }
    function modifyBorderColor(rgb, theme) {
        if (theme.mode === 0) {
            return modifyLightSchemeColor(rgb, theme);
        }
        const poleFg = getFgPole(theme);
        const poleBg = getBgPole(theme);
        return modifyColorWithCache(
            rgb,
            {...theme, mode: 0},
            modifyBorderHSL,
            poleFg,
            poleBg
        );
    }
    function modifyShadowColor(rgb, filter) {
        return modifyBackgroundColor(rgb, filter);
    }
    function modifyGradientColor(rgb, filter) {
        return modifyBackgroundColor(rgb, filter);
    }

    function createTextStyle(config) {
        const lines = [];
        lines.push(
            '*:not(pre, pre *, code, .far, .fa, .glyphicon, [class*="vjs-"], .fab, .fa-github, .fas,
.material-icons, .icofont, .typcn, mu, [class*="mu-"], .glyphicon, .icon) {'
        );
        if (config.useFont && config.fontFamily) {
            lines.push(`  font-family: ${config.fontFamily} !important;`);
        }
        if (config.textStroke > 0) {
            lines.push(
                `  -webkit-text-stroke: ${config.textStroke}px !important;`
            );
            lines.push(`  text-stroke: ${config.textStroke}px !important;`);
        }
        lines.push("}");
        return lines.join("\n");
    }

    var FilterMode;
    (function (FilterMode) {
        FilterMode[(FilterMode["light"] = 0)] = "light";
        FilterMode[(FilterMode["dark"] = 1)] = "dark";
    })(FilterMode || (FilterMode = {}));
```

```
function getCSSFilterValue(config) {
    const filters = [];
    if (config.mode === FilterMode.dark) {
        filters.push("invert(100%) hue-rotate(180deg)");
    }
    if (config.brightness !== 100) {
        filters.push(`brightness(${config.brightness}%)`);
    }
    if (config.contrast !== 100) {
        filters.push(`contrast(${config.contrast}%)`);
    }
    if (config.grayscale !== 0) {
        filters.push(`grayscale(${config.grayscale}%)`);
    }
    if (config.sepia !== 0) {
        filters.push(`sepia(${config.sepia}%)`);
    }
    if (filters.length === 0) {
        return null;
    }
    return filters.join(" ");
}

function toSVGMatrix(matrix) {
    return matrix
        .slice(0, 4)
        .map((m) => m.map((m) => m.toFixed(3)).join(" "))
        .join(" ");
}
function getSVGFilterMatrixValue(config) {
    return toSVGMatrix(createFilterMatrix(config));
}

function hexify(number) {
    return (number < 16 ? "0" : "") + number.toString(16);
}
function generateUID() {
    if ("randomUUID" in crypto) {
        const uuid = crypto.randomUUID();
        return (
            uuid.substring(0, 8) +
            uuid.substring(9, 13) +
            uuid.substring(14, 18) +
            uuid.substring(19, 23) +
            uuid.substring(24)
        );
    }
    if ("getRandomValues" in crypto) {
        return Array.from(crypto.getRandomValues(new Uint8Array(16)))
            .map((x) => hexify(x))
            .join("");
    }
    return Math.floor(Math.random() * 2 ** 55).toString(36);
}

const resolvers$1 = new Map();
const rejectors = new Map();
async function bgFetch(request) {
    return new Promise((resolve, reject) => {
        const id = generateUID();
        resolvers$1.set(id, resolve);
        rejectors.set(id, reject);
        chrome.runtime.sendMessage({
            type: MessageTypeCStoBG.FETCH,
            data: request,
            id
        });
    });
}
```

```
chrome.runtime.onMessage.addListener(({type, data, error, id}) => {
    if (type === MessageTypeBGtoCS.FETCH_RESPONSE) {
        const resolve = resolvers$1.get(id);
        const reject = rejectors.get(id);
        resolvers$1.delete(id);
        rejectors.delete(id);
        if (error) {
            reject && reject(error);
        } else {
            resolve && resolve(data);
        }
    }
});

async function getOKResponse(url, mimeType, origin) {
    const response = await fetch(url, {
        cache: "force-cache",
        credentials: "omit",
        referrer: origin
    });
    if (
        mimeType &&
        !response.headers.get("Content-Type").startsWith(mimeType)
    ) {
        throw new Error(`Mime type mismatch when loading ${url}`);
    }
    if (!response.ok) {
        throw new Error(
            `Unable to load ${url} ${response.status} ${response.statusText}`
        );
    }
    return response;
}
async function loadAsDataURL(url, mimeType) {
    const response = await getOKResponse(url, mimeType);
    return await readResponseAsDataURL(response);
}
async function loadAsBlob(url, mimeType) {
    const response = await getOKResponse(url, mimeType);
    return await response.blob();
}
async function readResponseAsDataURL(response) {
    const blob = await response.blob();
    const dataURL = await new Promise((resolve) => {
        const reader = new FileReader();
        reader.onloadend = () => resolve(reader.result);
        reader.readAsDataURL(blob);
    });
    return dataURL;
}

const MAX_FRAME_DURATION = 1000 / 60;
class AsyncQueue {
    constructor() {
        this.queue = [];
        this.timerId = null;
    }
    addTask(task) {
        this.queue.push(task);
        this.scheduleFrame();
    }
    stop() {
        if (this.timerId !== null) {
            cancelAnimationFrame(this.timerId);
            this.timerId = null;
        }
        this.queue = [];
    }
    scheduleFrame() {
```

```
        if (this.timerId) {
            return;
        }
        this.timerId = requestAnimationFrame(() => {
            this.timerId = null;
            const start = Date.now();
            let cb;
            while ((cb = this.queue.shift())) {
                cb();
                if (Date.now() - start >= MAX_FRAME_DURATION) {
                    this.scheduleFrame();
                    break;
                }
            }
        });
    }
}

const imageManager = new AsyncQueue();
async function getImageDetails(url) {
    return new Promise(async (resolve, reject) => {
        var _a, _b;
        try {
            const dataURL = url.startsWith("data:")
                ? url
                : await getDataURL(url);
            const blob =
                (_a = tryConvertDataURLToBlobSync(dataURL)) !== null &&
                _a !== void 0
                    ? _a
                    : await loadAsBlob(url);
            let image;
            if (dataURL.startsWith("data:image/svg+xml")) {
                image = await loadImage(dataURL);
            } else {
                image =
                    (_b = await tryCreateImageBitmap(blob)) !== null &&
                    _b !== void 0
                        ? _b
                        : await loadImage(dataURL);
            }
            imageManager.addTask(() => {
                const analysis = analyzeImage(image);
                resolve({
                    src: url,
                    blob,
                    dataURL,
                    width: image.width,
                    height: image.height,
                    ...analysis
                });
            });
        } catch (error) {
            reject(error);
        }
    });
}
async function getDataURL(url) {
    const parsedURL = new URL(url);
    if (parsedURL.origin === location.origin) {
        return await loadAsDataURL(url);
    }
    return await bgFetch({url, responseType: "data-url"});
}
async function tryCreateImageBitmap(blob) {
    try {
        return await createImageBitmap(blob);
    } catch (err) {
        logWarn(
```

```
            `Unable to create image bitmap for type ${blob.type}: ${String(
                err
            )}`
        );
        return null;
    }
}
const INCOMPLETE_DOC_LOADING_IMAGE_LIMIT = 256;
let loadingImagesCount = 0;
async function loadImage(url) {
    return new Promise((resolve, reject) => {
        const image = new Image();
        image.onload = () => resolve(image);
        image.onerror = () => reject(`Unable to load image ${url}`);
        if (
            ++loadingImagesCount <= INCOMPLETE_DOC_LOADING_IMAGE_LIMIT ||
            isReadyStateComplete()
        ) {
            image.src = url;
        } else {
            addReadyStateCompleteListener(() => (image.src = url));
        }
    });
}
const MAX_ANALYSIS_PIXELS_COUNT = 32 * 32;
let canvas;
let context;
function createCanvas() {
    const maxWidth = MAX_ANALYSIS_PIXELS_COUNT;
    const maxHeight = MAX_ANALYSIS_PIXELS_COUNT;
    canvas = document.createElement("canvas");
    canvas.width = maxWidth;
    canvas.height = maxHeight;
    context = canvas.getContext("2d", {willReadFrequently: true});
    context.imageSmoothingEnabled = false;
}
function removeCanvas() {
    canvas = null;
    context = null;
}
const LARGE_IMAGE_PIXELS_COUNT = 512 * 512;
function analyzeImage(image) {
    if (!canvas) {
        createCanvas();
    }
    let sw;
    let sh;
    if (image instanceof HTMLImageElement) {
        sw = image.naturalWidth;
        sh = image.naturalHeight;
    } else {
        sw = image.width;
        sh = image.height;
    }
    if (sw === 0 || sh === 0) {
        return {
            isDark: false,
            isLight: false,
            isTransparent: false,
            isLarge: false,
            isTooLarge: false
        };
    }
    if (sw * sh > LARGE_IMAGE_PIXELS_COUNT) {
        return {
            isDark: false,
            isLight: false,
            isTransparent: false,
            isLarge: true
```

```
        };
    }
    const sourcePixelsCount = sw * sh;
    const k = Math.min(
        1,
        Math.sqrt(MAX_ANALYSIS_PIXELS_COUNT / sourcePixelsCount)
    );
    const width = Math.ceil(sw * k);
    const height = Math.ceil(sh * k);
    context.clearRect(0, 0, width, height);
    context.drawImage(image, 0, 0, sw, sh, 0, 0, width, height);
    const imageData = context.getImageData(0, 0, width, height);
    const d = imageData.data;
    const TRANSPARENT_ALPHA_THRESHOLD = 0.05;
    const DARK_LIGHTNESS_THRESHOLD = 0.4;
    const LIGHT_LIGHTNESS_THRESHOLD = 0.7;
    let transparentPixelsCount = 0;
    let darkPixelsCount = 0;
    let lightPixelsCount = 0;
    let i, x, y;
    let r, g, b, a;
    let l;
    for (y = 0; y < height; y++) {
        for (x = 0; x < width; x++) {
            i = 4 * (y * width + x);
            r = d[i + 0];
            g = d[i + 1];
            b = d[i + 2];
            a = d[i + 3];
            if (a / 255 < TRANSPARENT_ALPHA_THRESHOLD) {
                transparentPixelsCount++;
            } else {
                l = getSRGBLightness(r, g, b);
                if (l < DARK_LIGHTNESS_THRESHOLD) {
                    darkPixelsCount++;
                }
                if (l > LIGHT_LIGHTNESS_THRESHOLD) {
                    lightPixelsCount++;
                }
            }
        }
    }
    const totalPixelsCount = width * height;
    const opaquePixelsCount = totalPixelsCount - transparentPixelsCount;
    const DARK_IMAGE_THRESHOLD = 0.7;
    const LIGHT_IMAGE_THRESHOLD = 0.7;
    const TRANSPARENT_IMAGE_THRESHOLD = 0.1;
    return {
        isDark: darkPixelsCount / opaquePixelsCount >= DARK_IMAGE_THRESHOLD,
        isLight:
            lightPixelsCount / opaquePixelsCount >= LIGHT_IMAGE_THRESHOLD,
        isTransparent:
            transparentPixelsCount / totalPixelsCount >=
            TRANSPARENT_IMAGE_THRESHOLD,
        isLarge: false
    };
}
let isBlobURLSupported = null;
let canUseProxy = false;
let blobURLCheckRequested = false;
const blobURLCheckAwaiters = [];
document.addEventListener(
    "__darkreader__inlineScriptsAllowed",
    () => (canUseProxy = true),
    {once: true}
);
async function requestBlobURLCheck() {
    if (!canUseProxy) {
        return;
```

```
        }
        if (blobURLCheckRequested) {
            return await new Promise((resolve) =>
                blobURLCheckAwaiters.push(resolve)
            );
        }
        blobURLCheckRequested = true;
        await new Promise((resolve) => {
            document.addEventListener(
                "__darkreader__blobURLCheckResponse",
                (e) => {
                    isBlobURLSupported = e.detail.blobURLAllowed;
                    resolve();
                    blobURLCheckAwaiters.forEach((r) => r());
                    blobURLCheckAwaiters.splice(0);
                },
                {once: true}
            );
            document.dispatchEvent(
                new CustomEvent("__darkreader__blobURLCheckRequest")
            );
        });
    }
    function isBlobURLCheckResultReady() {
        return isBlobURLSupported != null || !canUseProxy;
    }
    function onCSPError(err) {
        if (err.blockedURI === "blob") {
            isBlobURLSupported = false;
            document.removeEventListener("securitypolicyviolation", onCSPError);
        }
    }
    document.addEventListener("securitypolicyviolation", onCSPError);
    const objectURLs = new Set();
    function getFilteredImageURL({dataURL, width, height}, theme) {
        if (dataURL.startsWith("data:image/svg+xml")) {
            dataURL = escapeXML(dataURL);
        }
        const matrix = getSVGFilterMatrixValue(theme);
        const svg = [
            `<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" width="${width}"
height="${height}">`,
            "<defs>",
            '<filter id="darkreader-image-filter">',
            `<feColorMatrix type="matrix" values="${matrix}" />`,
            "</filter>",
            "</defs>",
            `<image width="${width}" height="${height}" filter="url(#darkreader-image-filter)"
xlink:href="${dataURL}" />`,
            "</svg>"
        ].join("");
        if (!isBlobURLSupported) {
            return `${btoa(svg)}`;
        }
        const bytes = new Uint8Array(svg.length);
        for (let i = 0; i < svg.length; i++) {
            bytes[i] = svg.charCodeAt(i);
        }
        const blob = new Blob([bytes], {type: "image/svg+xml"});
        const objectURL = URL.createObjectURL(blob);
        objectURLs.add(objectURL);
        return objectURL;
    }
    const xmlEscapeChars = {
        "<": "&lt;",
        ">": "&gt;",
        "&": "&amp;",
        "'": "&apos;",
        '"': "&quot;"
```

```
};
function escapeXML(str) {
    return str.replace(/[<>&'"]/g, (c) => {
        var _a;
        return (_a = xmlEscapeChars[c]) !== null && _a !== void 0 ? _a : c;
    });
}
const dataURLBlobURLs = new Map();
function tryConvertDataURLToBlobSync(dataURL) {
    const colonIndex = dataURL.indexOf(":");
    const semicolonIndex = dataURL.indexOf(";", colonIndex + 1);
    const commaIndex = dataURL.indexOf(",", semicolonIndex + 1);
    const encoding = dataURL
        .substring(semicolonIndex + 1, commaIndex)
        .toLocaleLowerCase();
    const mediaType = dataURL.substring(colonIndex + 1, semicolonIndex);
    if (encoding !== "base64" || !mediaType) {
        return null;
    }
    const characters = atob(dataURL.substring(commaIndex + 1));
    const bytes = new Uint8Array(characters.length);
    for (let i = 0; i < characters.length; i++) {
        bytes[i] = characters.charCodeAt(i);
    }
    return new Blob([bytes], {type: mediaType});
}
async function tryConvertDataURLToBlobURL(dataURL) {
    if (!isBlobURLSupported) {
        return null;
    }
    let blobURL = dataURLBlobURLs.get(dataURL);
    if (blobURL) {
        return blobURL;
    }
    let blob = tryConvertDataURLToBlobSync(dataURL);
    if (!blob) {
        const response = await fetch(dataURL);
        blob = await response.blob();
    }
    blobURL = URL.createObjectURL(blob);
    dataURLBlobURLs.set(dataURL, blobURL);
    return blobURL;
}
function cleanImageProcessingCache() {
    imageManager && imageManager.stop();
    removeCanvas();
    objectURLs.forEach((u) => URL.revokeObjectURL(u));
    objectURLs.clear();
    dataURLBlobURLs.forEach((u) => URL.revokeObjectURL(u));
    dataURLBlobURLs.clear();
}

const gradientLength = "gradient".length;
const conicGradient = "conic-";
const conicGradientLength = conicGradient.length;
const radialGradient = "radial-";
const linearGradient = "linear-";
function parseGradient(value) {
    const result = [];
    let index = 0;
    let startIndex = conicGradient.length;
    while ((index = value.indexOf("gradient", startIndex)) !== -1) {
        let typeGradient;
        [linearGradient, radialGradient, conicGradient].find(
            (possibleType) => {
                if (index - possibleType.length >= 0) {
                    const possibleGradient = value.substring(
                        index - possibleType.length,
                        index
```

```
                    );
                    if (possibleGradient === possibleType) {
                        if (
                            value.slice(
                                index - possibleType.length - 10,
                                index - possibleType.length - 1
                            ) === "repeating"
                        ) {
                            typeGradient = `repeating-${possibleType}gradient`;
                            return true;
                        }
                        if (
                            value.slice(
                                index - possibleType.length - 8,
                                index - possibleType.length - 1
                            ) === "-webkit"
                        ) {
                            typeGradient = `-webkit-${possibleType}gradient`;
                            return true;
                        }
                        typeGradient = `${possibleType}gradient`;
                        return true;
                    }
                }
            }
        );
        if (!typeGradient) {
            break;
        }
        const {start, end} = getParenthesesRange(
            value,
            index + gradientLength
        );
        const match = value.substring(start + 1, end - 1);
        startIndex = end + 1 + conicGradientLength;
        result.push({
            typeGradient,
            match,
            offset: typeGradient.length + 2,
            index: index - typeGradient.length + gradientLength,
            hasComma: true
        });
    }
    if (result.length) {
        result[result.length - 1].hasComma = false;
    }
    return result;
}

function getPriority(ruleStyle, property) {
    return Boolean(ruleStyle && ruleStyle.getPropertyPriority(property));
}
function getModifiableCSSDeclaration(
    property,
    value,
    rule,
    variablesStore,
    ignoreImageSelectors,
    isCancelled
) {
    if (property.startsWith("--")) {
        const modifier = getVariableModifier(
            variablesStore,
            property,
            value,
            rule,
            ignoreImageSelectors,
            isCancelled
        );
```

```
    if (modifier) {
        return {
            property,
            value: modifier,
            important: getPriority(rule.style, property),
            sourceValue: value
        };
    }
} else if (value.includes("var(")) {
    const modifier = getVariableDependantModifier(
        variablesStore,
        property,
        value
    );
    if (modifier) {
        return {
            property,
            value: modifier,
            important: getPriority(rule.style, property),
            sourceValue: value
        };
    }
} else if (property === "color-scheme") {
    return null;
} else if (
    (property.includes("color") &&
        property !== "-webkit-print-color-adjust") ||
    property === "fill" ||
    property === "stroke" ||
    property === "stop-color"
) {
    const modifier = getColorModifier(property, value, rule);
    if (modifier) {
        return {
            property,
            value: modifier,
            important: getPriority(rule.style, property),
            sourceValue: value
        };
    }
} else if (
    property === "background-image" ||
    property === "list-style-image"
) {
    const modifier = getBgImageModifier(
        value,
        rule,
        ignoreImageSelectors,
        isCancelled
    );
    if (modifier) {
        return {
            property,
            value: modifier,
            important: getPriority(rule.style, property),
            sourceValue: value
        };
    }
} else if (property.includes("shadow")) {
    const modifier = getShadowModifier(value);
    if (modifier) {
        return {
            property,
            value: modifier,
            important: getPriority(rule.style, property),
            sourceValue: value
        };
    }
}
```

```
    return null;
}
function joinSelectors(...selectors) {
    return selectors.filter(Boolean).join(", ");
}
function getModifiedUserAgentStyle(theme, isIFrame, styleSystemControls) {
    const lines = [];
    if (!isIFrame) {
        lines.push("html {");
        lines.push(
            `    background-color: ${modifyBackgroundColor(
                {r: 255, g: 255, b: 255},
                theme
            )} !important;`
        );
        lines.push("}");
    }
    if (isCSSColorSchemePropSupported) {
        lines.push("html {");
        lines.push(
            `    color-scheme: ${
                theme.mode === 1 ? "dark" : "dark light"
            } !important;`
        );
        lines.push("}");
    }
    const bgSelectors = joinSelectors(
        isIFrame ? "" : "html, body",
        styleSystemControls ? "input, textarea, select, button, dialog" : ""
    );
    if (bgSelectors) {
        lines.push(`${bgSelectors} {`);
        lines.push(
            `    background-color: ${modifyBackgroundColor(
                {r: 255, g: 255, b: 255},
                theme
            )};`
        );
        lines.push("}");
    }
    lines.push(
        `${joinSelectors(
            "html, body",
            styleSystemControls ? "input, textarea, select, button" : ""
        )} {`
    );
    lines.push(
        `    border-color: ${modifyBorderColor(
            {r: 76, g: 76, b: 76},
            theme
        )};`
    );
    lines.push(
        `    color: ${modifyForegroundColor({r: 0, g: 0, b: 0}, theme)};`
    );
    lines.push("}");
    lines.push("a {");
    lines.push(
        `    color: ${modifyForegroundColor({r: 0, g: 64, b: 255}, theme)};`
    );
    lines.push("}");
    lines.push("table {");
    lines.push(
        `    border-color: ${modifyBorderColor(
            {r: 128, g: 128, b: 128},
            theme
        )};`
    );
    lines.push("}");
```

```
    lines.push("::placeholder {");
    lines.push(
        `    color: ${modifyForegroundColor(
            {r: 169, g: 169, b: 169},
            theme
        )};`
    );
    lines.push("}");
    lines.push("input:-webkit-autofill,");
    lines.push("textarea:-webkit-autofill,");
    lines.push("select:-webkit-autofill {");
    lines.push(
        `    background-color: ${modifyBackgroundColor(
            {r: 250, g: 255, b: 189},
            theme
        )} !important;`
    );
    lines.push(
        `    color: ${modifyForegroundColor(
            {r: 0, g: 0, b: 0},
            theme
        )} !important;`
    );
    lines.push("}");
    if (theme.scrollbarColor) {
        lines.push(getModifiedScrollbarStyle(theme));
    }
    if (theme.selectionColor) {
        lines.push(getModifiedSelectionStyle(theme));
    }
    return lines.join("\n");
}
function getSelectionColor(theme) {
    let backgroundColorSelection;
    let foregroundColorSelection;
    if (theme.selectionColor === "auto") {
        backgroundColorSelection = modifyBackgroundColor(
            {r: 0, g: 96, b: 212},
            {...theme, grayscale: 0}
        );
        foregroundColorSelection = modifyForegroundColor(
            {r: 255, g: 255, b: 255},
            {...theme, grayscale: 0}
        );
    } else {
        const rgb = parseColorWithCache(theme.selectionColor);
        const hsl = rgbToHSL(rgb);
        backgroundColorSelection = theme.selectionColor;
        if (hsl.l < 0.5) {
            foregroundColorSelection = "#FFF";
        } else {
            foregroundColorSelection = "#000";
        }
    }
    return {backgroundColorSelection, foregroundColorSelection};
}
function getModifiedSelectionStyle(theme) {
    const lines = [];
    const modifiedSelectionColor = getSelectionColor(theme);
    const backgroundColorSelection =
        modifiedSelectionColor.backgroundColorSelection;
    const foregroundColorSelection =
        modifiedSelectionColor.foregroundColorSelection;
    ["::selection", "::-moz-selection"].forEach((selection) => {
        lines.push(`${selection} {`);
        lines.push(
            `    background-color: ${backgroundColorSelection} !important;`
        );
        lines.push(`    color: ${foregroundColorSelection} !important;`);
```

```
        lines.push("}");
    });
    return lines.join("\n");
}
function getModifiedScrollbarStyle(theme) {
    const lines = [];
    let colorTrack;
    let colorIcons;
    let colorThumb;
    let colorThumbHover;
    let colorThumbActive;
    let colorCorner;
    if (theme.scrollbarColor === "auto") {
        colorTrack = modifyBackgroundColor({r: 241, g: 241, b: 241}, theme);
        colorIcons = modifyForegroundColor({r: 96, g: 96, b: 96}, theme);
        colorThumb = modifyBackgroundColor({r: 176, g: 176, b: 176}, theme);
        colorThumbHover = modifyBackgroundColor(
            {r: 144, g: 144, b: 144},
            theme
        );
        colorThumbActive = modifyBackgroundColor(
            {r: 96, g: 96, b: 96},
            theme
        );
        colorCorner = modifyBackgroundColor(
            {r: 255, g: 255, b: 255},
            theme
        );
    } else {
        const rgb = parseColorWithCache(theme.scrollbarColor);
        const hsl = rgbToHSL(rgb);
        const isLight = hsl.l > 0.5;
        const lighten = (lighter) => ({
            ...hsl,
            l: clamp(hsl.l + lighter, 0, 1)
        });
        const darken = (darker) => ({
            ...hsl,
            l: clamp(hsl.l - darker, 0, 1)
        });
        colorTrack = hslToString(darken(0.4));
        colorIcons = hslToString(isLight ? darken(0.4) : lighten(0.4));
        colorThumb = hslToString(hsl);
        colorThumbHover = hslToString(lighten(0.1));
        colorThumbActive = hslToString(lighten(0.2));
        colorCorner = hslToString(darken(0.5));
    }
    lines.push("::-webkit-scrollbar {");
    lines.push(`    background-color: ${colorTrack};`);
    lines.push(`    color: ${colorIcons};`);
    lines.push("}");
    lines.push("::-webkit-scrollbar-thumb {");
    lines.push(`    background-color: ${colorThumb};`);
    lines.push("}");
    lines.push("::-webkit-scrollbar-thumb:hover {");
    lines.push(`    background-color: ${colorThumbHover};`);
    lines.push("}");
    lines.push("::-webkit-scrollbar-thumb:active {");
    lines.push(`    background-color: ${colorThumbActive};`);
    lines.push("}");
    lines.push("::-webkit-scrollbar-corner {");
    lines.push(`    background-color: ${colorCorner};`);
    lines.push("}");
    return lines.join("\n");
}
function getModifiedFallbackStyle(filter, {strict}) {
    const factory = defaultFallbackFactory;
    return factory(filter, {strict});
}
```

```
function defaultFallbackFactory(filter, {strict}) {
    const lines = [];
    const isMicrosoft = ["microsoft.com", "docs.microsoft.com"].includes(
        location.hostname
    );
    lines.push(
        `html, body, ${
            strict
                ? `body :not(iframe)${
                      isMicrosoft
                          ? ':not(div[style^="position:absolute;top:0;left:-"]'
                          : ""
                  }`
                : "body > :not(iframe)"
        } {`
    );
    lines.push(
        `    background-color: ${modifyBackgroundColor(
            {r: 255, g: 255, b: 255},
            filter
        )} !important;`
    );
    lines.push(
        `    border-color: ${modifyBorderColor(
            {r: 64, g: 64, b: 64},
            filter
        )} !important;`
    );
    lines.push(
        `    color: ${modifyForegroundColor(
            {r: 0, g: 0, b: 0},
            filter
        )} !important;`
    );
    lines.push("}");
    return lines.join("\n");
}
const unparsableColors = new Set([
    "inherit",
    "transparent",
    "initial",
    "currentcolor",
    "none",
    "unset"
]);
function getColorModifier(prop, value, rule) {
    if (unparsableColors.has(value.toLowerCase())) {
        return value;
    }
    const rgb = parseColorWithCache(value);
    if (!rgb) {
        return null;
    }
    if (prop.includes("background")) {
        if (
            (rule.style.webkitMaskImage &&
                rule.style.webkitMaskImage !== "none") ||
            (rule.style.webkitMask &&
                !rule.style.webkitMask.startsWith("none")) ||
            (rule.style.mask && rule.style.mask !== "none") ||
            (rule.style.getPropertyValue("mask-image") &&
                rule.style.getPropertyValue("mask-image") !== "none")
        ) {
            return (filter) => modifyForegroundColor(rgb, filter);
        }
        return (filter) => modifyBackgroundColor(rgb, filter);
    }
    if (prop.includes("border") || prop.includes("outline")) {
        return (filter) => modifyBorderColor(rgb, filter);
```

```
    }
    return (filter) => modifyForegroundColor(rgb, filter);
}
const imageDetailsCache = new Map();
const awaitingForImageLoading = new Map();
function shouldIgnoreImage(selectorText, selectors) {
    if (!selectorText || selectors.length === 0) {
        return false;
    }
    if (selectors.some((s) => s === "*")) {
        return true;
    }
    const ruleSelectors = selectorText.split(/,\s*/g);
    for (let i = 0; i < selectors.length; i++) {
        const ignoredSelector = selectors[i];
        if (ruleSelectors.some((s) => s === ignoredSelector)) {
            return true;
        }
    }
    return false;
}
function getBgImageModifier(
    value,
    rule,
    ignoreImageSelectors,
    isCancelled
) {
    try {
        const gradients = parseGradient(value);
        const urls = getMatches(cssURLRegex, value);
        if (urls.length === 0 && gradients.length === 0) {
            return value;
        }
        const getIndices = (matches) => {
            let index = 0;
            return matches.map((match) => {
                const valueIndex = value.indexOf(match, index);
                index = valueIndex + match.length;
                return {match, index: valueIndex};
            });
        };
        const matches = gradients
            .map((i) => ({type: "gradient", ...i}))
            .concat(
                getIndices(urls).map((i) => ({
                    type: "url",
                    offset: 0,
                    ...i
                }))
            )
            .sort((a, b) => (a.index > b.index ? 1 : -1));
        const getGradientModifier = (gradient) => {
            const {typeGradient, match, hasComma} = gradient;
            const partsRegex =
                /([^\(\),]+(\([^\(\)]*(\([^\(\)]*\)*[^\(\)]*)?\))?([^\(\), ]|( (?!calc)))*),?/g;
            const colorStopRegex =
                /^(from|color-stop|to)\(([^\(\)]*?,\s*)?(.*?)\)$/;
            const parts = getMatches(partsRegex, match, 1).map((part) => {
                part = part.trim();
                let rgb = parseColorWithCache(part);
                if (rgb) {
                    return (filter) => modifyGradientColor(rgb, filter);
                }
                const space = part.lastIndexOf(" ");
                rgb = parseColorWithCache(part.substring(0, space));
                if (rgb) {
                    return (filter) =>
                        `${modifyGradientColor(
                            rgb,
```

```
                    filter
                )} ${part.substring(space + 1)}`;
            }
            const colorStopMatch = part.match(colorStopRegex);
            if (colorStopMatch) {
                rgb = parseColorWithCache(colorStopMatch[3]);
                if (rgb) {
                    return (filter) =>
                        `${colorStopMatch[1]}(${
                            colorStopMatch[2]
                                ? `${colorStopMatch[2]}, `
                                : ""
                        }${modifyGradientColor(rgb, filter)})`;
                }
            }
            return () => part;
        });
        return (filter) => {
            return `${typeGradient}(${parts
                .map((modify) => modify(filter))
                .join(", ")})${hasComma ? ", " : ""}`;
        };
    };
    const getURLModifier = (urlValue) => {
        var _a;
        if (
            shouldIgnoreImage(rule.selectorText, ignoreImageSelectors)
        ) {
            return null;
        }
        let url = getCSSURLValue(urlValue);
        const isURLEmpty = url.length === 0;
        const {parentStyleSheet} = rule;
        const baseURL =
            parentStyleSheet && parentStyleSheet.href
                ? getCSSBaseBath(parentStyleSheet.href)
                : ((_a = parentStyleSheet.ownerNode) === null ||
                  _a === void 0
                      ? void 0
                      : _a.baseURI) || location.origin;
        url = getAbsoluteURL(baseURL, url);
        return async (filter) => {
            if (isURLEmpty) {
                return "url('')";
            }
            let imageDetails = null;
            if (imageDetailsCache.has(url)) {
                imageDetails = imageDetailsCache.get(url);
            } else {
                try {
                    if (!isBlobURLCheckResultReady()) {
                        await requestBlobURLCheck();
                    }
                    if (awaitingForImageLoading.has(url)) {
                        const awaiters =
                            awaitingForImageLoading.get(url);
                        imageDetails = await new Promise((resolve) =>
                            awaiters.push(resolve)
                        );
                        if (!imageDetails) {
                            return null;
                        }
                    } else {
                        awaitingForImageLoading.set(url, []);
                        imageDetails = await getImageDetails(url);
                        imageDetailsCache.set(url, imageDetails);
                        awaitingForImageLoading
                            .get(url)
                            .forEach((resolve) =>
```

```
                            resolve(imageDetails)
                        );
                    awaitingForImageLoading.delete(url);
                }
                if (isCancelled()) {
                    return null;
                }
            } catch (err) {
                logWarn(err);
                if (awaitingForImageLoading.has(url)) {
                    awaitingForImageLoading
                        .get(url)
                        .forEach((resolve) => resolve(null));
                    awaitingForImageLoading.delete(url);
                }
            }
        }
        if (imageDetails) {
            const bgImageValue = getBgImageValue(
                imageDetails,
                filter
            );
            if (bgImageValue) {
                return bgImageValue;
            }
        }
        if (url.startsWith("data:")) {
            const blobURL = await tryConvertDataURLToBlobURL(url);
            if (blobURL) {
                return `url("${blobURL}")`;
            }
        }
        return `url("${url}")`;
    };
};
const getBgImageValue = (imageDetails, filter) => {
    const {isDark, isLight, isTransparent, isLarge, width} =
        imageDetails;
    let result;
    const logSrc = imageDetails.src.startsWith("data:")
        ? "data:"
        : imageDetails.src;
    if (isLarge) {
        logInfo(`Not modifying too large image ${logSrc}`);
        result = null;
    } else if (
        isDark &&
        isTransparent &&
        filter.mode === 1 &&
        width > 2
    ) {
        logInfo(`Inverting dark image ${logSrc}`);
        const inverted = getFilteredImageURL(imageDetails, {
            ...filter,
            sepia: clamp(filter.sepia + 10, 0, 100)
        });
        result = `url("${inverted}")`;
    } else if (isLight && !isTransparent && filter.mode === 1) {
        logInfo(`Dimming light image ${logSrc}`);
        const dimmed = getFilteredImageURL(imageDetails, filter);
        result = `url("${dimmed}")`;
    } else if (filter.mode === 0 && isLight) {
        logInfo(`Applying filter to image ${logSrc}`);
        const filtered = getFilteredImageURL(imageDetails, {
            ...filter,
            brightness: clamp(filter.brightness - 10, 5, 200),
            sepia: clamp(filter.sepia + 10, 0, 100)
        });
        result = `url("${filtered}")`;
```

```
        } else {
            logInfo(`Not modifying the image ${logSrc}`);
            result = null;
        }
        return result;
    };
    const modifiers = [];
    let matchIndex = 0;
    let prevHasComma = false;
    matches.forEach(
        ({type, match, index, typeGradient, hasComma, offset}, i) => {
            const matchStart = index;
            const prefixStart = matchIndex;
            const matchEnd = matchStart + match.length + offset;
            matchIndex = matchEnd;
            if (prefixStart !== matchStart) {
                if (prevHasComma) {
                    modifiers.push(() => {
                        let betweenValue = value.substring(
                            prefixStart,
                            matchStart
                        );
                        if (betweenValue[0] === ",") {
                            betweenValue = betweenValue.substring(1);
                        }
                        return betweenValue;
                    });
                } else {
                    modifiers.push(() =>
                        value.substring(prefixStart, matchStart)
                    );
                }
            }
            prevHasComma = hasComma || false;
            if (type === "url") {
                modifiers.push(getURLModifier(match));
            } else if (type === "gradient") {
                modifiers.push(
                    getGradientModifier({
                        match,
                        index,
                        typeGradient: typeGradient,
                        hasComma: hasComma || false,
                        offset
                    })
                );
            }
            if (i === matches.length - 1) {
                modifiers.push(() => value.substring(matchEnd));
            }
        }
    );
    return (filter) => {
        const results = modifiers
            .filter(Boolean)
            .map((modify) => modify(filter));
        if (results.some((r) => r instanceof Promise)) {
            return Promise.all(results).then((asyncResults) => {
                return asyncResults.filter(Boolean).join("");
            });
        }
        const combinedResult = results.join("");
        if (combinedResult.endsWith(", initial")) {
            return combinedResult.slice(0, -9);
        }
        return combinedResult;
    };
} catch (err) {
    return null;
```

```
    }
}
function getShadowModifierWithInfo(value) {
    try {
        let index = 0;
        const colorMatches = getMatches(
            /(^|\s)(?!calc)([a-z]+\(.+?\)|#[0-9a-f]+|[a-z]+)(.*?(inset|outset)?($|,))/gi,
            value,
            2
        );
        let notParsed = 0;
        const modifiers = colorMatches.map((match, i) => {
            const prefixIndex = index;
            const matchIndex = value.indexOf(match, index);
            const matchEnd = matchIndex + match.length;
            index = matchEnd;
            const rgb = parseColorWithCache(match);
            if (!rgb) {
                notParsed++;
                return () => value.substring(prefixIndex, matchEnd);
            }
            return (filter) =>
                `${value.substring(
                    prefixIndex,
                    matchIndex
                )}${modifyShadowColor(rgb, filter)}${
                    i === colorMatches.length - 1
                        ? value.substring(matchEnd)
                        : ""
                }`;
        });
        return (filter) => {
            const modified = modifiers
                .map((modify) => modify(filter))
                .join("");
            return {
                matchesLength: colorMatches.length,
                unparseableMatchesLength: notParsed,
                result: modified
            };
        };
    } catch (err) {
        return null;
    }
}
function getShadowModifier(value) {
    const shadowModifier = getShadowModifierWithInfo(value);
    if (!shadowModifier) {
        return null;
    }
    return (theme) => shadowModifier(theme).result;
}
function getVariableModifier(
    variablesStore,
    prop,
    value,
    rule,
    ignoredImgSelectors,
    isCancelled
) {
    return variablesStore.getModifierForVariable({
        varName: prop,
        sourceValue: value,
        rule,
        ignoredImgSelectors,
        isCancelled
    });
}
function getVariableDependantModifier(variablesStore, prop, value) {
```

```
    return variablesStore.getModifierForVarDependant(prop, value);
}
function cleanModificationCache() {
    clearColorModificationCache();
    imageDetailsCache.clear();
    cleanImageProcessingCache();
    awaitingForImageLoading.clear();
}

const VAR_TYPE_BGCOLOR = 1 << 0;
const VAR_TYPE_TEXTCOLOR = 1 << 1;
const VAR_TYPE_BORDERCOLOR = 1 << 2;
const VAR_TYPE_BGIMG = 1 << 3;
class VariablesStore {
    constructor() {
        this.varTypes = new Map();
        this.rulesQueue = [];
        this.inlineStyleQueue = [];
        this.definedVars = new Set();
        this.varRefs = new Map();
        this.unknownColorVars = new Set();
        this.unknownBgVars = new Set();
        this.undefinedVars = new Set();
        this.initialVarTypes = new Map();
        this.changedTypeVars = new Set();
        this.typeChangeSubscriptions = new Map();
        this.unstableVarValues = new Map();
    }
    clear() {
        this.varTypes.clear();
        this.rulesQueue.splice(0);
        this.inlineStyleQueue.splice(0);
        this.definedVars.clear();
        this.varRefs.clear();
        this.unknownColorVars.clear();
        this.unknownBgVars.clear();
        this.undefinedVars.clear();
        this.initialVarTypes.clear();
        this.changedTypeVars.clear();
        this.typeChangeSubscriptions.clear();
        this.unstableVarValues.clear();
    }
    isVarType(varName, typeNum) {
        return (
            this.varTypes.has(varName) &&
            (this.varTypes.get(varName) & typeNum) > 0
        );
    }
    addRulesForMatching(rules) {
        this.rulesQueue.push(rules);
    }
    addInlineStyleForMatching(style) {
        this.inlineStyleQueue.push(style);
    }
    matchVariablesAndDependents() {
        if (
            this.rulesQueue.length === 0 &&
            this.inlineStyleQueue.length === 0
        ) {
            return;
        }
        this.changedTypeVars.clear();
        this.initialVarTypes = new Map(this.varTypes);
        this.collectRootVariables();
        this.collectVariablesAndVarDep();
        this.collectRootVarDependents();
        this.varRefs.forEach((refs, v) => {
            refs.forEach((r) => {
                if (this.varTypes.has(v)) {
```

```
                this.resolveVariableType(r, this.varTypes.get(v));
            }
        });
    });
    this.unknownColorVars.forEach((v) => {
        if (this.unknownBgVars.has(v)) {
            this.unknownColorVars.delete(v);
            this.unknownBgVars.delete(v);
            this.resolveVariableType(v, VAR_TYPE_BGCOLOR);
        } else if (
            this.isVarType(
                v,
                VAR_TYPE_BGCOLOR |
                    VAR_TYPE_TEXTCOLOR |
                    VAR_TYPE_BORDERCOLOR
            )
        ) {
            this.unknownColorVars.delete(v);
        } else {
            this.undefinedVars.add(v);
        }
    });
    this.unknownBgVars.forEach((v) => {
        const hasColor =
            this.findVarRef(v, (ref) => {
                return (
                    this.unknownColorVars.has(ref) ||
                    this.isVarType(
                        ref,
                        VAR_TYPE_TEXTCOLOR | VAR_TYPE_BORDERCOLOR
                    )
                );
            }) != null;
        if (hasColor) {
            this.iterateVarRefs(v, (ref) => {
                this.resolveVariableType(ref, VAR_TYPE_BGCOLOR);
            });
        } else if (
            this.isVarType(v, VAR_TYPE_BGCOLOR | VAR_TYPE_BGIMG)
        ) {
            this.unknownBgVars.delete(v);
        } else {
            this.undefinedVars.add(v);
        }
    });
    this.changedTypeVars.forEach((varName) => {
        if (this.typeChangeSubscriptions.has(varName)) {
            this.typeChangeSubscriptions
                .get(varName)
                .forEach((callback) => {
                    callback();
                });
        }
    });
    this.changedTypeVars.clear();
}
getModifierForVariable(options) {
    return (theme) => {
        const {
            varName,
            sourceValue,
            rule,
            ignoredImgSelectors,
            isCancelled
        } = options;
        const getDeclarations = () => {
            const declarations = [];
            const addModifiedValue = (
                typeNum,
```

```
    varNameWrapper,
    colorModifier
) => {
    if (!this.isVarType(varName, typeNum)) {
        return;
    }
    const property = varNameWrapper(varName);
    let modifiedValue;
    if (isVarDependant(sourceValue)) {
        if (isConstructedColorVar(sourceValue)) {
            let value = insertVarValues(
                sourceValue,
                this.unstableVarValues
            );
            if (!value) {
                value =
                    typeNum === VAR_TYPE_BGCOLOR
                        ? "#ffffff"
                        : "#000000";
            }
            modifiedValue = colorModifier(value, theme);
        } else {
            modifiedValue = replaceCSSVariablesNames(
                sourceValue,
                (v) => varNameWrapper(v),
                (fallback) => colorModifier(fallback, theme)
            );
        }
    } else {
        modifiedValue = colorModifier(sourceValue, theme);
    }
    declarations.push({
        property,
        value: modifiedValue
    });
};
addModifiedValue(
    VAR_TYPE_BGCOLOR,
    wrapBgColorVariableName,
    tryModifyBgColor
);
addModifiedValue(
    VAR_TYPE_TEXTCOLOR,
    wrapTextColorVariableName,
    tryModifyTextColor
);
addModifiedValue(
    VAR_TYPE_BORDERCOLOR,
    wrapBorderColorVariableName,
    tryModifyBorderColor
);
if (this.isVarType(varName, VAR_TYPE_BGIMG)) {
    const property = wrapBgImgVariableName(varName);
    let modifiedValue = sourceValue;
    if (isVarDependant(sourceValue)) {
        modifiedValue = replaceCSSVariablesNames(
            sourceValue,
            (v) => wrapBgColorVariableName(v),
            (fallback) => tryModifyBgColor(fallback, theme)
        );
    }
    const bgModifier = getBgImageModifier(
        modifiedValue,
        rule,
        ignoredImgSelectors,
        isCancelled
    );
    modifiedValue =
        typeof bgModifier === "function"
```

```
                        ? bgModifier(theme)
                        : bgModifier;
                declarations.push({
                    property,
                    value: modifiedValue
                });
            }
            return declarations;
        };
        const callbacks = new Set();
        const addListener = (onTypeChange) => {
            const callback = () => {
                const decs = getDeclarations();
                onTypeChange(decs);
            };
            callbacks.add(callback);
            this.subscribeForVarTypeChange(varName, callback);
        };
        const removeListeners = () => {
            callbacks.forEach((callback) => {
                this.unsubscribeFromVariableTypeChanges(
                    varName,
                    callback
                );
            });
        };
        return {
            declarations: getDeclarations(),
            onTypeChange: {addListener, removeListeners}
        };
    };
}
getModifierForVarDependant(property, sourceValue) {
    if (sourceValue.match(/^\s*(rgb|hsl)a?\(/)) {
        const isBg = property.startsWith("background");
        const isText = isTextColorProperty(property);
        return (theme) => {
            let value = insertVarValues(
                sourceValue,
                this.unstableVarValues
            );
            if (!value) {
                value = isBg ? "#ffffff" : "#000000";
            }
            const modifier = isBg
                ? tryModifyBgColor
                : isText
                  ? tryModifyTextColor
                  : tryModifyBorderColor;
            return modifier(value, theme);
        };
    }
    if (property === "background-color") {
        return (theme) => {
            return replaceCSSVariablesNames(
                sourceValue,
                (v) => wrapBgColorVariableName(v),
                (fallback) => tryModifyBgColor(fallback, theme)
            );
        };
    }
    if (isTextColorProperty(property)) {
        return (theme) => {
            return replaceCSSVariablesNames(
                sourceValue,
                (v) => wrapTextColorVariableName(v),
                (fallback) => tryModifyTextColor(fallback, theme)
            );
        };
```

```
}
if (
    property === "background" ||
    property === "background-image" ||
    property === "box-shadow"
) {
    return (theme) => {
        const unknownVars = new Set();
        const modify = () => {
            const variableReplaced = replaceCSSVariablesNames(
                sourceValue,
                (v) => {
                    if (this.isVarType(v, VAR_TYPE_BGCOLOR)) {
                        return wrapBgColorVariableName(v);
                    }
                    if (this.isVarType(v, VAR_TYPE_BGIMG)) {
                        return wrapBgImgVariableName(v);
                    }
                    unknownVars.add(v);
                    return v;
                },
                (fallback) => tryModifyBgColor(fallback, theme)
            );
            if (property === "box-shadow") {
                const shadowModifier =
                    getShadowModifierWithInfo(variableReplaced);
                const modifiedShadow = shadowModifier(theme);
                if (
                    modifiedShadow.unparseableMatchesLength !==
                    modifiedShadow.matchesLength
                ) {
                    return modifiedShadow.result;
                }
            }
            return variableReplaced;
        };
        const modified = modify();
        if (unknownVars.size > 0) {
            const isFallbackResolved = modified.match(
                /^var\(.*?, var\(--darkreader-bg--.*\)\)$/
            );
            if (isFallbackResolved) {
                return modified;
            }
            return new Promise((resolve) => {
                const firstUnknownVar = unknownVars
                    .values()
                    .next().value;
                const callback = () => {
                    this.unsubscribeFromVariableTypeChanges(
                        firstUnknownVar,
                        callback
                    );
                    const newValue = modify();
                    resolve(newValue);
                };
                this.subscribeForVarTypeChange(
                    firstUnknownVar,
                    callback
                );
            });
        }
        return modified;
    };
}
if (
    property.startsWith("border") ||
    property.startsWith("outline")
) {
```

```
        return (theme) => {
            return replaceCSSVariablesNames(
                sourceValue,
                (v) => wrapBorderColorVariableName(v),
                (fallback) => tryModifyBorderColor(fallback, theme)
            );
        };
    }
    return null;
}
subscribeForVarTypeChange(varName, callback) {
    if (!this.typeChangeSubscriptions.has(varName)) {
        this.typeChangeSubscriptions.set(varName, new Set());
    }
    const rootStore = this.typeChangeSubscriptions.get(varName);
    if (!rootStore.has(callback)) {
        rootStore.add(callback);
    }
}
unsubscribeFromVariableTypeChanges(varName, callback) {
    if (this.typeChangeSubscriptions.has(varName)) {
        this.typeChangeSubscriptions.get(varName).delete(callback);
    }
}
collectVariablesAndVarDep() {
    this.rulesQueue.forEach((rules) => {
        iterateCSSRules(rules, (rule) => {
            if (rule.style) {
                this.collectVarsFromCSSDeclarations(rule.style);
            }
        });
    });
    this.inlineStyleQueue.forEach((style) => {
        this.collectVarsFromCSSDeclarations(style);
    });
    this.rulesQueue.splice(0);
    this.inlineStyleQueue.splice(0);
}
collectVarsFromCSSDeclarations(style) {
    iterateCSSDeclarations(style, (property, value) => {
        if (isVariable(property)) {
            this.inspectVariable(property, value);
        }
        if (isVarDependant(value)) {
            this.inspectVarDependant(property, value);
        }
    });
}
shouldProcessRootVariables() {
    var _a;
    return (
        this.rulesQueue.length > 0 &&
        ((_a = document.documentElement.getAttribute("style")) ===
            null || _a === void 0
            ? void 0
            : _a.includes("--"))
    );
}
collectRootVariables() {
    if (!this.shouldProcessRootVariables()) {
        return;
    }
    iterateCSSDeclarations(
        document.documentElement.style,
        (property, value) => {
            if (isVariable(property)) {
                this.inspectVariable(property, value);
            }
        }
```

```
    );
}
inspectVariable(varName, value) {
    this.unstableVarValues.set(varName, value);
    if (isVarDependant(value) && isConstructedColorVar(value)) {
        this.unknownColorVars.add(varName);
        this.definedVars.add(varName);
    }
    if (this.definedVars.has(varName)) {
        return;
    }
    this.definedVars.add(varName);
    const isColor =
        rawValueRegex.test(value) || parseColorWithCache(value);
    if (isColor) {
        this.unknownColorVars.add(varName);
    } else if (
        value.includes("url(") ||
        value.includes("linear-gradient(") ||
        value.includes("radial-gradient(")
    ) {
        this.resolveVariableType(varName, VAR_TYPE_BGIMG);
    }
}
resolveVariableType(varName, typeNum) {
    const initialType = this.initialVarTypes.get(varName) || 0;
    const currentType = this.varTypes.get(varName) || 0;
    const newType = currentType | typeNum;
    this.varTypes.set(varName, newType);
    if (newType !== initialType || this.undefinedVars.has(varName)) {
        this.changedTypeVars.add(varName);
        this.undefinedVars.delete(varName);
    }
    this.unknownColorVars.delete(varName);
    this.unknownBgVars.delete(varName);
}
collectRootVarDependents() {
    if (!this.shouldProcessRootVariables()) {
        return;
    }
    iterateCSSDeclarations(
        document.documentElement.style,
        (property, value) => {
            if (isVarDependant(value)) {
                this.inspectVarDependant(property, value);
            }
        }
    );
}
inspectVarDependant(property, value) {
    if (isVariable(property)) {
        this.iterateVarDeps(value, (ref) => {
            if (!this.varRefs.has(property)) {
                this.varRefs.set(property, new Set());
            }
            this.varRefs.get(property).add(ref);
        });
    } else if (
        property === "background-color" ||
        property === "box-shadow"
    ) {
        this.iterateVarDeps(value, (v) =>
            this.resolveVariableType(v, VAR_TYPE_BGCOLOR)
        );
    } else if (isTextColorProperty(property)) {
        this.iterateVarDeps(value, (v) =>
            this.resolveVariableType(v, VAR_TYPE_TEXTCOLOR)
        );
    } else if (
```

```
        property.startsWith("border") ||
        property.startsWith("outline")
    ) {
        this.iterateVarDeps(value, (v) =>
            this.resolveVariableType(v, VAR_TYPE_BORDERCOLOR)
        );
    } else if (
        property === "background" ||
        property === "background-image"
    ) {
        this.iterateVarDeps(value, (v) => {
            if (this.isVarType(v, VAR_TYPE_BGCOLOR | VAR_TYPE_BGIMG)) {
                return;
            }
            const isBgColor =
                this.findVarRef(v, (ref) => {
                    return (
                        this.unknownColorVars.has(ref) ||
                        this.isVarType(
                            ref,
                            VAR_TYPE_TEXTCOLOR | VAR_TYPE_BORDERCOLOR
                        )
                    );
                }) != null;
            this.iterateVarRefs(v, (ref) => {
                if (isBgColor) {
                    this.resolveVariableType(ref, VAR_TYPE_BGCOLOR);
                } else {
                    this.unknownBgVars.add(ref);
                }
            });
        });
    }
}
iterateVarDeps(value, iterator) {
    const varDeps = new Set();
    iterateVarDependencies(value, (v) => varDeps.add(v));
    varDeps.forEach((v) => iterator(v));
}
findVarRef(varName, iterator, stack = new Set()) {
    if (stack.has(varName)) {
        return null;
    }
    stack.add(varName);
    const result = iterator(varName);
    if (result) {
        return varName;
    }
    const refs = this.varRefs.get(varName);
    if (!refs || refs.size === 0) {
        return null;
    }
    for (const ref of refs) {
        const found = this.findVarRef(ref, iterator, stack);
        if (found) {
            return found;
        }
    }
    return null;
}
iterateVarRefs(varName, iterator) {
    this.findVarRef(varName, (ref) => {
        iterator(ref);
        return false;
    });
}
setOnRootVariableChange(callback) {
    this.onRootVariableDefined = callback;
}
```

```
    putRootVars(styleElement, theme) {
        const sheet = styleElement.sheet;
        if (sheet.cssRules.length > 0) {
            sheet.deleteRule(0);
        }
        const declarations = new Map();
        iterateCSSDeclarations(
            document.documentElement.style,
            (property, value) => {
                if (isVariable(property)) {
                    if (this.isVarType(property, VAR_TYPE_BGCOLOR)) {
                        declarations.set(
                            wrapBgColorVariableName(property),
                            tryModifyBgColor(value, theme)
                        );
                    }
                    if (this.isVarType(property, VAR_TYPE_TEXTCOLOR)) {
                        declarations.set(
                            wrapTextColorVariableName(property),
                            tryModifyTextColor(value, theme)
                        );
                    }
                    if (this.isVarType(property, VAR_TYPE_BORDERCOLOR)) {
                        declarations.set(
                            wrapBorderColorVariableName(property),
                            tryModifyBorderColor(value, theme)
                        );
                    }
                    this.subscribeForVarTypeChange(
                        property,
                        this.onRootVariableDefined
                    );
                }
            }
        );
        const cssLines = [];
        cssLines.push(":root {");
        for (const [property, value] of declarations) {
            cssLines.push(`    ${property}: ${value};`);
        }
        cssLines.push("}");
        const cssText = cssLines.join("\n");
        sheet.insertRule(cssText);
    }
}
const variablesStore = new VariablesStore();
function getVariableRange(input, searchStart = 0) {
    const start = input.indexOf("var(", searchStart);
    if (start >= 0) {
        const range = getParenthesesRange(input, start + 3);
        if (range) {
            return {start, end: range.end};
        }
    }
    return null;
}
function getVariablesMatches(input) {
    const ranges = [];
    let i = 0;
    let range;
    while ((range = getVariableRange(input, i))) {
        const {start, end} = range;
        ranges.push({start, end, value: input.substring(start, end)});
        i = range.end + 1;
    }
    return ranges;
}
function replaceVariablesMatches(input, replacer) {
    const matches = getVariablesMatches(input);
```

```
    const matchesCount = matches.length;
    if (matchesCount === 0) {
        return input;
    }
    const inputLength = input.length;
    const replacements = matches.map((m) => replacer(m.value));
    const parts = [];
    parts.push(input.substring(0, matches[0].start));
    for (let i = 0; i < matchesCount; i++) {
        parts.push(replacements[i]);
        const start = matches[i].end;
        const end =
            i < matchesCount - 1 ? matches[i + 1].start : inputLength;
        parts.push(input.substring(start, end));
    }
    return parts.join("");
}
function getVariableNameAndFallback(match) {
    const commaIndex = match.indexOf(",");
    let name;
    let fallback;
    if (commaIndex >= 0) {
        name = match.substring(4, commaIndex).trim();
        fallback = match.substring(commaIndex + 1, match.length - 1).trim();
    } else {
        name = match.substring(4, match.length - 1).trim();
        fallback = "";
    }
    return {name, fallback};
}
function replaceCSSVariablesNames(value, nameReplacer, fallbackReplacer) {
    const matchReplacer = (match) => {
        const {name, fallback} = getVariableNameAndFallback(match);
        const newName = nameReplacer(name);
        if (!fallback) {
            return `var(${newName})`;
        }
        let newFallback;
        if (isVarDependant(fallback)) {
            newFallback = replaceCSSVariablesNames(
                fallback,
                nameReplacer,
                fallbackReplacer
            );
        } else if (fallbackReplacer) {
            newFallback = fallbackReplacer(fallback);
        } else {
            newFallback = fallback;
        }
        return `var(${newName}, ${newFallback})`;
    };
    return replaceVariablesMatches(value, matchReplacer);
}
function iterateVarDependencies(value, iterator) {
    replaceCSSVariablesNames(value, (varName) => {
        iterator(varName);
        return varName;
    });
}
function wrapBgColorVariableName(name) {
    return `--darkreader-bg${name}`;
}
function wrapTextColorVariableName(name) {
    return `--darkreader-text${name}`;
}
function wrapBorderColorVariableName(name) {
    return `--darkreader-border${name}`;
}
function wrapBgImgVariableName(name) {
```

```
    return `--darkreader-bgimg${name}`;
}
function isVariable(property) {
    return property.startsWith("--");
}
function isVarDependant(value) {
    return value.includes("var(");
}
function isConstructedColorVar(value) {
    return value.match(/^\s*(rgb|hsl)a?\(/);
}
function isTextColorProperty(property) {
    return (
        property === "color" ||
        property === "caret-color" ||
        property === "-webkit-text-fill-color"
    );
}
const rawValueRegex = /^\d{1,3}, ?\d{1,3}, ?\d{1,3}$/;
function parseRawValue(color) {
    if (rawValueRegex.test(color)) {
        const splitted = color.split(",");
        let resultInRGB = "rgb(";
        splitted.forEach((number) => {
            resultInRGB += `${number.trim()}, `;
        });
        resultInRGB = resultInRGB.substring(0, resultInRGB.length - 2);
        resultInRGB += ")";
        return {isRaw: true, color: resultInRGB};
    }
    return {isRaw: false, color: color};
}
function handleRawValue(color, theme, modifyFunction) {
    const {isRaw, color: newColor} = parseRawValue(color);
    const rgb = parseColorWithCache(newColor);
    if (rgb) {
        const outputColor = modifyFunction(rgb, theme);
        if (isRaw) {
            const outputInRGB = parseColorWithCache(outputColor);
            return outputInRGB
                ? `${outputInRGB.r}, ${outputInRGB.g}, ${outputInRGB.b}`
                : outputColor;
        }
        return outputColor;
    }
    return newColor;
}
function tryModifyBgColor(color, theme) {
    return handleRawValue(color, theme, modifyBackgroundColor);
}
function tryModifyTextColor(color, theme) {
    return handleRawValue(color, theme, modifyForegroundColor);
}
function tryModifyBorderColor(color, theme) {
    return handleRawValue(color, theme, modifyBorderColor);
}
function insertVarValues(source, varValues, stack = new Set()) {
    let containsUnresolvedVar = false;
    const matchReplacer = (match) => {
        const {name, fallback} = getVariableNameAndFallback(match);
        if (stack.has(name)) {
            containsUnresolvedVar = true;
            return null;
        }
        stack.add(name);
        const varValue = varValues.get(name) || fallback;
        let inserted = null;
        if (varValue) {
            if (isVarDependant(varValue)) {
```

```
                inserted = insertVarValues(varValue, varValues, stack);
            } else {
                inserted = varValue;
            }
        }
        if (!inserted) {
            containsUnresolvedVar = true;
            return null;
        }
        return inserted;
    };
    const replaced = replaceVariablesMatches(source, matchReplacer);
    if (containsUnresolvedVar) {
        return null;
    }
    return replaced;
}

const overrides$1 = {
    "background-color": {
        customProp: "--darkreader-inline-bgcolor",
        cssProp: "background-color",
        dataAttr: "data-darkreader-inline-bgcolor"
    },
    "background-image": {
        customProp: "--darkreader-inline-bgimage",
        cssProp: "background-image",
        dataAttr: "data-darkreader-inline-bgimage"
    },
    "border-color": {
        customProp: "--darkreader-inline-border",
        cssProp: "border-color",
        dataAttr: "data-darkreader-inline-border"
    },
    "border-bottom-color": {
        customProp: "--darkreader-inline-border-bottom",
        cssProp: "border-bottom-color",
        dataAttr: "data-darkreader-inline-border-bottom"
    },
    "border-left-color": {
        customProp: "--darkreader-inline-border-left",
        cssProp: "border-left-color",
        dataAttr: "data-darkreader-inline-border-left"
    },
    "border-right-color": {
        customProp: "--darkreader-inline-border-right",
        cssProp: "border-right-color",
        dataAttr: "data-darkreader-inline-border-right"
    },
    "border-top-color": {
        customProp: "--darkreader-inline-border-top",
        cssProp: "border-top-color",
        dataAttr: "data-darkreader-inline-border-top"
    },
    "box-shadow": {
        customProp: "--darkreader-inline-boxshadow",
        cssProp: "box-shadow",
        dataAttr: "data-darkreader-inline-boxshadow"
    },
    "color": {
        customProp: "--darkreader-inline-color",
        cssProp: "color",
        dataAttr: "data-darkreader-inline-color"
    },
    "fill": {
        customProp: "--darkreader-inline-fill",
        cssProp: "fill",
        dataAttr: "data-darkreader-inline-fill"
    },
```

```
    "stroke": {
        customProp: "--darkreader-inline-stroke",
        cssProp: "stroke",
        dataAttr: "data-darkreader-inline-stroke"
    },
    "outline-color": {
        customProp: "--darkreader-inline-outline",
        cssProp: "outline-color",
        dataAttr: "data-darkreader-inline-outline"
    },
    "stop-color": {
        customProp: "--darkreader-inline-stopcolor",
        cssProp: "stop-color",
        dataAttr: "data-darkreader-inline-stopcolor"
    }
};
const shorthandOverrides = {
    background: {
        customProp: "--darkreader-inline-bg",
        cssProp: "background",
        dataAttr: "data-darkreader-inline-bg"
    }
};
const overridesList = Object.values(overrides$1);
const normalizedPropList = {};
overridesList.forEach(
    ({cssProp, customProp}) => (normalizedPropList[customProp] = cssProp)
);
const INLINE_STYLE_ATTRS = [
    "style",
    "fill",
    "stop-color",
    "stroke",
    "bgcolor",
    "color"
];
const INLINE_STYLE_SELECTOR = INLINE_STYLE_ATTRS.map(
    (attr) => `[${attr}]`
).join(", ");
function getInlineOverrideStyle() {
    const allOverrides = overridesList.concat(
        Object.values(shorthandOverrides)
    );
    return allOverrides
        .map(({dataAttr, customProp, cssProp}) => {
            return [
                `[${dataAttr}] {`,
                `  ${cssProp}: var(${customProp}) !important;`,
                "}"
            ].join("\n");
        })
        .join("\n");
}
function getInlineStyleElements(root) {
    const results = [];
    if (root instanceof Element && root.matches(INLINE_STYLE_SELECTOR)) {
        results.push(root);
    }
    if (
        root instanceof Element ||
        (isShadowDomSupported && root instanceof ShadowRoot) ||
        root instanceof Document
    ) {
        push(results, root.querySelectorAll(INLINE_STYLE_SELECTOR));
    }
    return results;
}
const treeObservers = new Map();
const attrObservers = new Map();
```

```
function watchForInlineStyles(elementStyleDidChange, shadowRootDiscovered) {
    deepWatchForInlineStyles(
        document,
        elementStyleDidChange,
        shadowRootDiscovered
    );
    iterateShadowHosts(document.documentElement, (host) => {
        deepWatchForInlineStyles(
            host.shadowRoot,
            elementStyleDidChange,
            shadowRootDiscovered
        );
    });
}
function deepWatchForInlineStyles(
    root,
    elementStyleDidChange,
    shadowRootDiscovered
) {
    if (treeObservers.has(root)) {
        treeObservers.get(root).disconnect();
        attrObservers.get(root).disconnect();
    }
    const discoveredNodes = new WeakSet();
    function discoverNodes(node) {
        getInlineStyleElements(node).forEach((el) => {
            if (discoveredNodes.has(el)) {
                return;
            }
            discoveredNodes.add(el);
            elementStyleDidChange(el);
        });
        iterateShadowHosts(node, (n) => {
            if (discoveredNodes.has(node)) {
                return;
            }
            discoveredNodes.add(node);
            shadowRootDiscovered(n.shadowRoot);
            deepWatchForInlineStyles(
                n.shadowRoot,
                elementStyleDidChange,
                shadowRootDiscovered
            );
        });
        variablesStore.matchVariablesAndDependents();
    }
    const treeObserver = createOptimizedTreeObserver(root, {
        onMinorMutations: ({additions}) => {
            additions.forEach((added) => discoverNodes(added));
        },
        onHugeMutations: () => {
            discoverNodes(root);
        }
    });
    treeObservers.set(root, treeObserver);
    let attemptCount = 0;
    let start = null;
    const ATTEMPTS_INTERVAL = getDuration({seconds: 10});
    const RETRY_TIMEOUT = getDuration({seconds: 2});
    const MAX_ATTEMPTS_COUNT = 50;
    let cache = [];
    let timeoutId = null;
    const handleAttributeMutations = throttle((mutations) => {
        const handledTargets = new Set();
        mutations.forEach((m) => {
            const target = m.target;
            if (handledTargets.has(target)) {
                return;
            }
```

```
            if (INLINE_STYLE_ATTRS.includes(m.attributeName)) {
                handledTargets.add(target);
                elementStyleDidChange(target);
            }
        });
        variablesStore.matchVariablesAndDependents();
    });
    const attrObserver = new MutationObserver((mutations) => {
        if (timeoutId) {
            cache.push(...mutations);
            return;
        }
        attemptCount++;
        const now = Date.now();
        if (start == null) {
            start = now;
        } else if (attemptCount >= MAX_ATTEMPTS_COUNT) {
            if (now - start < ATTEMPTS_INTERVAL) {
                timeoutId = setTimeout(() => {
                    start = null;
                    attemptCount = 0;
                    timeoutId = null;
                    const attributeCache = cache;
                    cache = [];
                    handleAttributeMutations(attributeCache);
                }, RETRY_TIMEOUT);
                cache.push(...mutations);
                return;
            }
            start = now;
            attemptCount = 1;
        }
        handleAttributeMutations(mutations);
    });
    attrObserver.observe(root, {
        attributes: true,
        attributeFilter: INLINE_STYLE_ATTRS.concat(
            overridesList.map(({dataAttr}) => dataAttr)
        ),
        subtree: true
    });
    attrObservers.set(root, attrObserver);
}
function stopWatchingForInlineStyles() {
    treeObservers.forEach((o) => o.disconnect());
    attrObservers.forEach((o) => o.disconnect());
    treeObservers.clear();
    attrObservers.clear();
}
const inlineStyleCache = new WeakMap();
const filterProps = [
    "brightness",
    "contrast",
    "grayscale",
    "sepia",
    "mode"
];
function getInlineStyleCacheKey(el, theme) {
    return INLINE_STYLE_ATTRS.map(
        (attr) => `${attr}="${el.getAttribute(attr)}"`
    )
        .concat(filterProps.map((prop) => `${prop}="${theme[prop]}"`))
        .join(" ");
}
function shouldIgnoreInlineStyle(element, selectors) {
    for (let i = 0, len = selectors.length; i < len; i++) {
        const ingnoredSelector = selectors[i];
        if (element.matches(ingnoredSelector)) {
            return true;
```

```
        }
    }
    return false;
}
function overrideInlineStyle(
    element,
    theme,
    ignoreInlineSelectors,
    ignoreImageSelectors
) {
    var _a;
    const cacheKey = getInlineStyleCacheKey(element, theme);
    if (cacheKey === inlineStyleCache.get(element)) {
        return;
    }
    const unsetProps = new Set(Object.keys(overrides$1));
    function setCustomProp(targetCSSProp, modifierCSSProp, cssVal) {
        const mod = getModifiableCSSDeclaration(
            modifierCSSProp,
            cssVal,
            {style: element.style},
            variablesStore,
            ignoreImageSelectors,
            null
        );
        if (!mod) {
            return;
        }
        function setStaticValue(value) {
            var _a;
            const {customProp, dataAttr} =
                (_a = overrides$1[targetCSSProp]) !== null && _a !== void 0
                    ? _a
                    : shorthandOverrides[targetCSSProp];
            element.style.setProperty(customProp, value);
            if (!element.hasAttribute(dataAttr)) {
                element.setAttribute(dataAttr, "");
            }
            unsetProps.delete(targetCSSProp);
        }
        function setVarDeclaration(mod) {
            let prevDeclarations = [];
            function setProps(declarations) {
                prevDeclarations.forEach(({property}) => {
                    element.style.removeProperty(property);
                });
                declarations.forEach(({property, value}) => {
                    if (!(value instanceof Promise)) {
                        element.style.setProperty(property, value);
                    }
                });
                prevDeclarations = declarations;
            }
            setProps(mod.declarations);
            mod.onTypeChange.addListener(setProps);
        }
        function setAsyncValue(promise) {
            promise.then((value) => {
                if (
                    value &&
                    targetCSSProp === "background" &&
                    value.startsWith("var(--darkreader-bg--")
                ) {
                    setStaticValue(value);
                }
            });
        }
        const value =
            typeof mod.value === "function" ? mod.value(theme) : mod.value;
```

```
    if (typeof value === "string") {
        setStaticValue(value);
    } else if (value instanceof Promise) {
        setAsyncValue(value);
    } else if (typeof value === "object") {
        setVarDeclaration(value);
    }
}
if (ignoreInlineSelectors.length > 0) {
    if (shouldIgnoreInlineStyle(element, ignoreInlineSelectors)) {
        unsetProps.forEach((cssProp) => {
            element.removeAttribute(overrides$1[cssProp].dataAttr);
        });
        return;
    }
}
if (element.hasAttribute("bgcolor")) {
    let value = element.getAttribute("bgcolor");
    if (
        value.match(/^[0-9a-f]{3}$/i) ||
        value.match(/^[0-9a-f]{6}$/i)
    ) {
        value = `#${value}`;
    }
    setCustomProp("background-color", "background-color", value);
}
if (element.hasAttribute("color") && element.rel !== "mask-icon") {
    let value = element.getAttribute("color");
    if (
        value.match(/^[0-9a-f]{3}$/i) ||
        value.match(/^[0-9a-f]{6}$/i)
    ) {
        value = `#${value}`;
    }
    setCustomProp("color", "color", value);
}
if (element instanceof SVGElement) {
    if (element.hasAttribute("fill")) {
        const SMALL_SVG_LIMIT = 32;
        const value = element.getAttribute("fill");
        if (value !== "none") {
            if (!(element instanceof SVGTextElement)) {
                const handleSVGElement = () => {
                    const {width, height} =
                        element.getBoundingClientRect();
                    const isBg =
                        width > SMALL_SVG_LIMIT ||
                        height > SMALL_SVG_LIMIT;
                    setCustomProp(
                        "fill",
                        isBg ? "background-color" : "color",
                        value
                    );
                };
                if (isReadyStateComplete()) {
                    handleSVGElement();
                } else {
                    addReadyStateCompleteListener(handleSVGElement);
                }
            } else {
                setCustomProp("fill", "color", value);
            }
        }
    }
    if (element.hasAttribute("stop-color")) {
        setCustomProp(
            "stop-color",
            "background-color",
            element.getAttribute("stop-color")
```

```
            );
        }
    }
    if (element.hasAttribute("stroke")) {
        const value = element.getAttribute("stroke");
        setCustomProp(
            "stroke",
            element instanceof SVGLineElement ||
                element instanceof SVGTextElement
                ? "border-color"
                : "color",
            value
        );
    }
    element.style &&
        iterateCSSDeclarations(element.style, (property, value) => {
            if (property === "background-image" && value.includes("url")) {
                return;
            }
            if (
                overrides$1.hasOwnProperty(property) ||
                (property.startsWith("--") && !normalizedPropList[property])
            ) {
                setCustomProp(property, property, value);
            } else if (
                property === "background" &&
                value.includes("var(")
            ) {
                setCustomProp("background", "background", value);
            } else {
                const overriddenProp = normalizedPropList[property];
                if (
                    overriddenProp &&
                    !element.style.getPropertyValue(overriddenProp) &&
                    !element.hasAttribute(overriddenProp)
                ) {
                    if (
                        overriddenProp === "background-color" &&
                        element.hasAttribute("bgcolor")
                    ) {
                        return;
                    }
                    element.style.setProperty(property, "");
                }
            }
        });
    if (
        element.style &&
        element instanceof SVGTextElement &&
        element.style.fill
    ) {
        setCustomProp(
            "fill",
            "color",
            element.style.getPropertyValue("fill")
        );
    }
    if (
        (_a = element.getAttribute("style")) === null || _a === void 0
            ? void 0
            : _a.includes("--")
    ) {
        variablesStore.addInlineStyleForMatching(element.style);
    }
    forEach(unsetProps, (cssProp) => {
        element.removeAttribute(overrides$1[cssProp].dataAttr);
    });
    inlineStyleCache.set(element, getInlineStyleCacheKey(element, theme));
}
```

```
const metaThemeColorName = "theme-color";
const metaThemeColorSelector = `meta[name="${metaThemeColorName}"]`;
let srcMetaThemeColor = null;
let observer = null;
function changeMetaThemeColor(meta, theme) {
    srcMetaThemeColor = srcMetaThemeColor || meta.content;
    const color = parseColorWithCache(srcMetaThemeColor);
    if (!color) {
        return;
    }
    meta.content = modifyBackgroundColor(color, theme);
}
function changeMetaThemeColorWhenAvailable(theme) {
    const meta = document.querySelector(metaThemeColorSelector);
    if (meta) {
        changeMetaThemeColor(meta, theme);
    } else {
        if (observer) {
            observer.disconnect();
        }
        observer = new MutationObserver((mutations) => {
            loop: for (let i = 0; i < mutations.length; i++) {
                const {addedNodes} = mutations[i];
                for (let j = 0; j < addedNodes.length; j++) {
                    const node = addedNodes[j];
                    if (
                        node instanceof HTMLMetaElement &&
                        node.name === metaThemeColorName
                    ) {
                        observer.disconnect();
                        observer = null;
                        changeMetaThemeColor(node, theme);
                        break loop;
                    }
                }
            }
        });
        observer.observe(document.head, {childList: true});
    }
}
function restoreMetaThemeColor() {
    if (observer) {
        observer.disconnect();
        observer = null;
    }
    const meta = document.querySelector(metaThemeColorSelector);
    if (meta && srcMetaThemeColor) {
        meta.content = srcMetaThemeColor;
    }
}

const themeCacheKeys = [
    "mode",
    "brightness",
    "contrast",
    "grayscale",
    "sepia",
    "darkSchemeBackgroundColor",
    "darkSchemeTextColor",
    "lightSchemeBackgroundColor",
    "lightSchemeTextColor"
];
function getThemeKey(theme) {
    let resultKey = "";
    themeCacheKeys.forEach((key) => {
        resultKey += `${key}:${theme[key]};`;
    });
    return resultKey;
```

```
}
const asyncQueue = createAsyncTasksQueue();
function createStyleSheetModifier() {
    let renderId = 0;
    const rulesTextCache = new Set();
    const rulesModCache = new Map();
    const varTypeChangeCleaners = new Set();
    let prevFilterKey = null;
    let hasNonLoadedLink = false;
    let wasRebuilt = false;
    function shouldRebuildStyle() {
        return hasNonLoadedLink && !wasRebuilt;
    }
    function modifySheet(options) {
        const rules = options.sourceCSSRules;
        const {
            theme,
            ignoreImageAnalysis,
            force,
            prepareSheet,
            isAsyncCancelled
        } = options;
        let rulesChanged = rulesModCache.size === 0;
        const notFoundCacheKeys = new Set(rulesModCache.keys());
        const themeKey = getThemeKey(theme);
        const themeChanged = themeKey !== prevFilterKey;
        if (hasNonLoadedLink) {
            wasRebuilt = true;
        }
        const modRules = [];
        iterateCSSRules(
            rules,
            (rule) => {
                let cssText = rule.cssText;
                let textDiffersFromPrev = false;
                notFoundCacheKeys.delete(cssText);
                if (rule.parentRule instanceof CSSMediaRule) {
                    cssText += `;${rule.parentRule.media.mediaText}`;
                }
                if (!rulesTextCache.has(cssText)) {
                    rulesTextCache.add(cssText);
                    textDiffersFromPrev = true;
                }
                if (textDiffersFromPrev) {
                    rulesChanged = true;
                } else {
                    modRules.push(rulesModCache.get(cssText));
                    return;
                }
                if (rule.style.all === "revert") {
                    return;
                }
                const modDecs = [];
                rule.style &&
                    iterateCSSDeclarations(
                        rule.style,
                        (property, value) => {
                            const mod = getModifiableCSSDeclaration(
                                property,
                                value,
                                rule,
                                variablesStore,
                                ignoreImageAnalysis,
                                isAsyncCancelled
                            );
                            if (mod) {
                                modDecs.push(mod);
                            }
                        }
```

```
            );
        let modRule = null;
        if (modDecs.length > 0) {
            const parentRule = rule.parentRule;
            modRule = {
                selector: rule.selectorText,
                declarations: modDecs,
                parentRule
            };
            modRules.push(modRule);
        }
        rulesModCache.set(cssText, modRule);
    },
    () => {
        hasNonLoadedLink = true;
    }
);
notFoundCacheKeys.forEach((key) => {
    rulesTextCache.delete(key);
    rulesModCache.delete(key);
});
prevFilterKey = themeKey;
if (!force && !rulesChanged && !themeChanged) {
    return;
}
renderId++;
function setRule(target, index, rule) {
    const {selector, declarations} = rule;
    let selectorText = selector;
    if (
        selector.startsWith(":is(") &&
        (selector.includes(":is()") ||
            selector.includes(":where()") ||
            (selector.includes(":where(") &&
                selector.includes(":-moz")))
    ) {
        selectorText = ".darkreader-unsupported-selector";
    }
    let ruleText = `${selectorText} {`;
    for (const dec of declarations) {
        const {property, value, important} = dec;
        if (value) {
            ruleText += ` ${property}: ${value}${
                important ? " !important" : ""
            };`;
        }
    }
    ruleText += " }";
    target.insertRule(ruleText, index);
}
const asyncDeclarations = new Map();
const varDeclarations = new Map();
let asyncDeclarationCounter = 0;
let varDeclarationCounter = 0;
const rootReadyGroup = {rule: null, rules: [], isGroup: true};
const groupRefs = new WeakMap();
function getGroup(rule) {
    if (rule == null) {
        return rootReadyGroup;
    }
    if (groupRefs.has(rule)) {
        return groupRefs.get(rule);
    }
    const group = {rule, rules: [], isGroup: true};
    groupRefs.set(rule, group);
    const parentGroup = getGroup(rule.parentRule);
    parentGroup.rules.push(group);
    return group;
}
```

```
varTypeChangeCleaners.forEach((clear) => clear());
varTypeChangeCleaners.clear();
modRules
    .filter((r) => r)
    .forEach(({selector, declarations, parentRule}) => {
        const group = getGroup(parentRule);
        const readyStyleRule = {
            selector,
            declarations: [],
            isGroup: false
        };
        const readyDeclarations = readyStyleRule.declarations;
        group.rules.push(readyStyleRule);
        function handleAsyncDeclaration(
            property,
            modified,
            important,
            sourceValue
        ) {
            const asyncKey = ++asyncDeclarationCounter;
            const asyncDeclaration = {
                property,
                value: null,
                important,
                asyncKey,
                sourceValue
            };
            readyDeclarations.push(asyncDeclaration);
            const currentRenderId = renderId;
            modified.then((asyncValue) => {
                if (
                    !asyncValue ||
                    isAsyncCancelled() ||
                    currentRenderId !== renderId
                ) {
                    return;
                }
                asyncDeclaration.value = asyncValue;
                asyncQueue.add(() => {
                    if (
                        isAsyncCancelled() ||
                        currentRenderId !== renderId
                    ) {
                        return;
                    }
                    rebuildAsyncRule(asyncKey);
                });
            });
        }
        function handleVarDeclarations(
            property,
            modified,
            important,
            sourceValue
        ) {
            const {declarations: varDecs, onTypeChange} = modified;
            const varKey = ++varDeclarationCounter;
            const currentRenderId = renderId;
            const initialIndex = readyDeclarations.length;
            let oldDecs = [];
            if (varDecs.length === 0) {
                const tempDec = {
                    property,
                    value: sourceValue,
                    important,
                    sourceValue,
                    varKey
                };
                readyDeclarations.push(tempDec);
```

```
            oldDecs = [tempDec];
        }
        varDecs.forEach((mod) => {
            if (mod.value instanceof Promise) {
                handleAsyncDeclaration(
                    mod.property,
                    mod.value,
                    important,
                    sourceValue
                );
            } else {
                const readyDec = {
                    property: mod.property,
                    value: mod.value,
                    important,
                    sourceValue,
                    varKey
                };
                readyDeclarations.push(readyDec);
                oldDecs.push(readyDec);
            }
        });
        onTypeChange.addListener((newDecs) => {
            if (
                isAsyncCancelled() ||
                currentRenderId !== renderId
            ) {
                return;
            }
            const readyVarDecs = newDecs.map((mod) => {
                return {
                    property: mod.property,
                    value: mod.value,
                    important,
                    sourceValue,
                    varKey
                };
            });
            const index = readyDeclarations.indexOf(
                oldDecs[0],
                initialIndex
            );
            readyDeclarations.splice(
                index,
                oldDecs.length,
                ...readyVarDecs
            );
            oldDecs = readyVarDecs;
            rebuildVarRule(varKey);
        });
        varTypeChangeCleaners.add(() =>
            onTypeChange.removeListeners()
        );
    }
    declarations.forEach(
        ({property, value, important, sourceValue}) => {
            if (typeof value === "function") {
                const modified = value(theme);
                if (modified instanceof Promise) {
                    handleAsyncDeclaration(
                        property,
                        modified,
                        important,
                        sourceValue
                    );
                } else if (property.startsWith("--")) {
                    handleVarDeclarations(
                        property,
                        modified,
```

```
                            important,
                            sourceValue
                        );
                    } else {
                        readyDeclarations.push({
                            property,
                            value: modified,
                            important,
                            sourceValue
                        });
                    }
                } else {
                    readyDeclarations.push({
                        property,
                        value,
                        important,
                        sourceValue
                    });
                }
            }
        );
    });
const sheet = prepareSheet();
function buildStyleSheet() {
    function createTarget(group, parent) {
        const {rule} = group;
        if (rule instanceof CSSMediaRule) {
            const {media} = rule;
            const index = parent.cssRules.length;
            parent.insertRule(
                `@media ${media.mediaText} {}`,
                index
            );
            return parent.cssRules[index];
        }
        return parent;
    }
    function iterateReadyRules(group, target, styleIterator) {
        group.rules.forEach((r) => {
            if (r.isGroup) {
                const t = createTarget(r, target);
                iterateReadyRules(r, t, styleIterator);
            } else {
                styleIterator(r, target);
            }
        });
    }
    iterateReadyRules(rootReadyGroup, sheet, (rule, target) => {
        const index = target.cssRules.length;
        rule.declarations.forEach(({asyncKey, varKey}) => {
            if (asyncKey != null) {
                asyncDeclarations.set(asyncKey, {
                    rule,
                    target,
                    index
                });
            }
            if (varKey != null) {
                varDeclarations.set(varKey, {rule, target, index});
            }
        });
        setRule(target, index, rule);
    });
}
function rebuildAsyncRule(key) {
    const {rule, target, index} = asyncDeclarations.get(key);
    target.deleteRule(index);
    setRule(target, index, rule);
    asyncDeclarations.delete(key);
```

```
        }
        function rebuildVarRule(key) {
            const {rule, target, index} = varDeclarations.get(key);
            target.deleteRule(index);
            setRule(target, index, rule);
        }
        buildStyleSheet();
    }
    return {modifySheet, shouldRebuildStyle};
}

const STYLE_SELECTOR = 'style, link[rel*="stylesheet" i]:not([disabled])';
function isFontsGoogleApiStyle(element) {
    if (!element.href) {
        return false;
    }
    try {
        const elementURL = new URL(element.href);
        return elementURL.hostname === "fonts.googleapis.com";
    } catch (err) {
        logInfo(`Couldn't construct ${element.href} as URL`);
        return false;
    }
}
const hostsBreakingOnSVGStyleOverride = ["www.onet.pl"];
function shouldManageStyle(element) {
    return (
        (element instanceof HTMLStyleElement ||
            (element instanceof SVGStyleElement &&
                !hostsBreakingOnSVGStyleOverride.includes(
                    location.hostname
                )) ||
            (element instanceof HTMLLinkElement &&
                Boolean(element.rel) &&
                element.rel.toLowerCase().includes("stylesheet") &&
                Boolean(element.href) &&
                !element.disabled &&
                true &&
                !isFontsGoogleApiStyle(element))) &&
        !element.classList.contains("darkreader") &&
        element.media.toLowerCase() !== "print" &&
        !element.classList.contains("stylus")
    );
}
function getManageableStyles(node, results = [], deep = true) {
    if (shouldManageStyle(node)) {
        results.push(node);
    } else if (
        node instanceof Element ||
        (isShadowDomSupported && node instanceof ShadowRoot) ||
        node === document
    ) {
        forEach(node.querySelectorAll(STYLE_SELECTOR), (style) =>
            getManageableStyles(style, results, false)
        );
        if (deep) {
            iterateShadowHosts(node, (host) =>
                getManageableStyles(host.shadowRoot, results, false)
            );
        }
    }
    return results;
}
const syncStyleSet = new WeakSet();
const corsStyleSet = new WeakSet();
let canOptimizeUsingProxy$1 = false;
document.addEventListener(
    "__darkreader__inlineScriptsAllowed",
    () => {
```

```
        canOptimizeUsingProxy$1 = true;
    },
    {once: true, passive: true}
);
let loadingLinkCounter = 0;
const rejectorsForLoadingLinks = new Map();
function cleanLoadingLinks() {
    rejectorsForLoadingLinks.clear();
}
function manageStyle(element, {update, loadingStart, loadingEnd}) {
    const prevStyles = [];
    let next = element;
    while (
        (next = next.nextElementSibling) &&
        next.matches(".darkreader")
    ) {
        prevStyles.push(next);
    }
    let corsCopy =
        prevStyles.find(
            (el) => el.matches(".darkreader--cors") && !corsStyleSet.has(el)
        ) || null;
    let syncStyle =
        prevStyles.find(
            (el) => el.matches(".darkreader--sync") && !syncStyleSet.has(el)
        ) || null;
    let corsCopyPositionWatcher = null;
    let syncStylePositionWatcher = null;
    let cancelAsyncOperations = false;
    let isOverrideEmpty = true;
    const sheetModifier = createStyleSheetModifier();
    const observer = new MutationObserver(() => {
        update();
    });
    const observerOptions = {
        attributes: true,
        childList: true,
        subtree: true,
        characterData: true
    };
    function containsCSSImport() {
        var _a;
        if (!(element instanceof HTMLStyleElement)) {
            return false;
        }
        const cssText = removeCSSComments(
            (_a = element.textContent) !== null && _a !== void 0 ? _a : ""
        ).trim();
        return cssText.match(cssImportRegex);
    }
    function hasImports(cssRules, checkCrossOrigin) {
        let result = false;
        if (cssRules) {
            let rule;
            cssRulesLoop: for (
                let i = 0, len = cssRules.length;
                i < len;
                i++
            ) {
                rule = cssRules[i];
                if (rule.href) {
                    if (checkCrossOrigin) {
                        if (
                            !rule.href.startsWith(
                                "https://fonts.googleapis.com/"
                            ) &&
                            rule.href.startsWith("http") &&
                            !rule.href.startsWith(location.origin)
                        ) {
```

```
                        result = true;
                        break cssRulesLoop;
                    }
                } else {
                    result = true;
                    break cssRulesLoop;
                }
            }
        }
    }
    return result;
}
function getRulesSync() {
    if (corsCopy) {
        return corsCopy.sheet.cssRules;
    }
    if (containsCSSImport()) {
        return null;
    }
    const cssRules = safeGetSheetRules();
    if (
        element instanceof HTMLLinkElement &&
        !isRelativeHrefOnAbsolutePath(element.href) &&
        hasImports(cssRules, false)
    ) {
        return null;
    }
    if (hasImports(cssRules, true)) {
        return null;
    }
    return cssRules;
}
function insertStyle() {
    if (corsCopy) {
        if (element.nextSibling !== corsCopy) {
            element.parentNode.insertBefore(
                corsCopy,
                element.nextSibling
            );
        }
        if (corsCopy.nextSibling !== syncStyle) {
            element.parentNode.insertBefore(
                syncStyle,
                corsCopy.nextSibling
            );
        }
    } else if (element.nextSibling !== syncStyle) {
        element.parentNode.insertBefore(syncStyle, element.nextSibling);
    }
}
function createSyncStyle() {
    syncStyle =
        element instanceof SVGStyleElement
            ? document.createElementNS(
                  "http://www.w3.org/2000/svg",
                  "style"
              )
            : document.createElement("style");
    syncStyle.classList.add("darkreader");
    syncStyle.classList.add("darkreader--sync");
    syncStyle.media = "screen";
    if (element.title) {
        syncStyle.title = element.title;
    }
    syncStyleSet.add(syncStyle);
}
let isLoadingRules = false;
let wasLoadingError = false;
const loadingLinkId = ++loadingLinkCounter;
```

```
async function getRulesAsync() {
    let cssText;
    let cssBasePath;
    if (element instanceof HTMLLinkElement) {
        let [cssRules, accessError] = getRulesOrError();
        if (
            (!cssRules && !accessError) ||
            isStillLoadingError(accessError)
        ) {
            try {
                logInfo(
                    `Linkelement ${loadingLinkId} is not loaded yet and thus will be await for`,
                    element
                );
                await linkLoading(element, loadingLinkId);
            } catch (err) {
                wasLoadingError = true;
            }
            if (cancelAsyncOperations) {
                return null;
            }
            [cssRules, accessError] = getRulesOrError();
        }
        if (cssRules) {
            if (!hasImports(cssRules, false)) {
                return cssRules;
            }
        }
        cssText = await loadText(element.href);
        cssBasePath = getCSSBaseBath(element.href);
        if (cancelAsyncOperations) {
            return null;
        }
    } else if (containsCSSImport()) {
        cssText = element.textContent.trim();
        cssBasePath = getCSSBaseBath(location.href);
    } else {
        return null;
    }
    if (cssText) {
        try {
            const fullCSSText = await replaceCSSImports(
                cssText,
                cssBasePath
            );
            corsCopy = createCORSCopy(element, fullCSSText);
        } catch (err) {}
        if (corsCopy) {
            corsCopyPositionWatcher = watchForNodePosition(
                corsCopy,
                "prev-sibling"
            );
            return corsCopy.sheet.cssRules;
        }
    }
    return null;
}
function details(options) {
    const rules = getRulesSync();
    if (!rules) {
        if (options.secondRound) {
            return null;
        }
        if (isLoadingRules || wasLoadingError) {
            return null;
        }
        isLoadingRules = true;
        loadingStart();
        getRulesAsync()
```

```
            .then((results) => {
                isLoadingRules = false;
                loadingEnd();
                if (results) {
                    update();
                }
            })
            .catch((err) => {
                isLoadingRules = false;
                loadingEnd();
            });
        return null;
    }
    return {rules};
}
let forceRenderStyle = false;
function render(theme, ignoreImageAnalysis) {
    const rules = getRulesSync();
    if (!rules) {
        return;
    }
    cancelAsyncOperations = false;
    function removeCSSRulesFromSheet(sheet) {
        if (!sheet) {
            return;
        }
        for (let i = sheet.cssRules.length - 1; i >= 0; i--) {
            sheet.deleteRule(i);
        }
    }
    function prepareOverridesSheet() {
        if (!syncStyle) {
            createSyncStyle();
        }
        syncStylePositionWatcher && syncStylePositionWatcher.stop();
        insertStyle();
        if (syncStyle.sheet == null) {
            syncStyle.textContent = "";
        }
        const sheet = syncStyle.sheet;
        removeCSSRulesFromSheet(sheet);
        if (syncStylePositionWatcher) {
            syncStylePositionWatcher.run();
        } else {
            syncStylePositionWatcher = watchForNodePosition(
                syncStyle,
                "prev-sibling",
                () => {
                    forceRenderStyle = true;
                    buildOverrides();
                }
            );
        }
        return syncStyle.sheet;
    }
    function buildOverrides() {
        const force = forceRenderStyle;
        forceRenderStyle = false;
        sheetModifier.modifySheet({
            prepareSheet: prepareOverridesSheet,
            sourceCSSRules: rules,
            theme,
            ignoreImageAnalysis,
            force,
            isAsyncCancelled: () => cancelAsyncOperations
        });
        isOverrideEmpty = syncStyle.sheet.cssRules.length === 0;
        if (sheetModifier.shouldRebuildStyle()) {
            addReadyStateCompleteListener(() => update());
```

```
        }
    }
    buildOverrides();
}
function getRulesOrError() {
    try {
        if (element.sheet == null) {
            return [null, null];
        }
        return [element.sheet.cssRules, null];
    } catch (err) {
        return [null, err];
    }
}
function isStillLoadingError(error) {
    return error && error.message && error.message.includes("loading");
}
function safeGetSheetRules() {
    const [cssRules, err] = getRulesOrError();
    if (err) {
        return null;
    }
    return cssRules;
}
function watchForSheetChanges() {
    watchForSheetChangesUsingProxy();
    if (!(canOptimizeUsingProxy$1 && element.sheet)) {
        watchForSheetChangesUsingRAF();
    }
}
let rulesChangeKey = null;
let rulesCheckFrameId = null;
function getRulesChangeKey() {
    const rules = safeGetSheetRules();
    return rules ? rules.length : null;
}
function didRulesKeyChange() {
    return getRulesChangeKey() !== rulesChangeKey;
}
function watchForSheetChangesUsingRAF() {
    rulesChangeKey = getRulesChangeKey();
    stopWatchingForSheetChangesUsingRAF();
    const checkForUpdate = () => {
        if (didRulesKeyChange()) {
            rulesChangeKey = getRulesChangeKey();
            update();
        }
        if (canOptimizeUsingProxy$1 && element.sheet) {
            stopWatchingForSheetChangesUsingRAF();
            return;
        }
        rulesCheckFrameId = requestAnimationFrame(checkForUpdate);
    };
    checkForUpdate();
}
function stopWatchingForSheetChangesUsingRAF() {
    cancelAnimationFrame(rulesCheckFrameId);
}
let areSheetChangesPending = false;
function onSheetChange() {
    canOptimizeUsingProxy$1 = true;
    stopWatchingForSheetChangesUsingRAF();
    if (areSheetChangesPending) {
        return;
    }
    function handleSheetChanges() {
        areSheetChangesPending = false;
        if (cancelAsyncOperations) {
            return;
```

```
        }
        update();
    }
    areSheetChangesPending = true;
    if (typeof queueMicrotask === "function") {
        queueMicrotask(handleSheetChanges);
    } else {
        requestAnimationFrame(handleSheetChanges);
    }
}
function watchForSheetChangesUsingProxy() {
    element.addEventListener(
        "__darkreader__updateSheet",
        onSheetChange,
        {passive: true}
    );
}
function stopWatchingForSheetChangesUsingProxy() {
    element.removeEventListener(
        "__darkreader__updateSheet",
        onSheetChange
    );
}
function stopWatchingForSheetChanges() {
    stopWatchingForSheetChangesUsingProxy();
    stopWatchingForSheetChangesUsingRAF();
}
function pause() {
    observer.disconnect();
    cancelAsyncOperations = true;
    corsCopyPositionWatcher && corsCopyPositionWatcher.stop();
    syncStylePositionWatcher && syncStylePositionWatcher.stop();
    stopWatchingForSheetChanges();
}
function destroy() {
    pause();
    removeNode(corsCopy);
    removeNode(syncStyle);
    loadingEnd();
    if (rejectorsForLoadingLinks.has(loadingLinkId)) {
        const reject = rejectorsForLoadingLinks.get(loadingLinkId);
        rejectorsForLoadingLinks.delete(loadingLinkId);
        reject && reject();
    }
}
function watch() {
    observer.observe(element, observerOptions);
    if (element instanceof HTMLStyleElement) {
        watchForSheetChanges();
    }
}
const maxMoveCount = 10;
let moveCount = 0;
function restore() {
    if (!syncStyle) {
        return;
    }
    moveCount++;
    if (moveCount > maxMoveCount) {
        return;
    }
    insertStyle();
    corsCopyPositionWatcher && corsCopyPositionWatcher.skip();
    syncStylePositionWatcher && syncStylePositionWatcher.skip();
    if (!isOverrideEmpty) {
        forceRenderStyle = true;
        update();
    }
}
```

```
    return {
        details,
        render,
        pause,
        destroy,
        watch,
        restore
    };
}
async function linkLoading(link, loadingId) {
    return new Promise((resolve, reject) => {
        const cleanUp = () => {
            link.removeEventListener("load", onLoad);
            link.removeEventListener("error", onError);
            rejectorsForLoadingLinks.delete(loadingId);
        };
        const onLoad = () => {
            cleanUp();
            resolve();
        };
        const onError = () => {
            cleanUp();
            reject(
                `Linkelement ${loadingId} couldn't be loaded. ${link.href}`
            );
        };
        rejectorsForLoadingLinks.set(loadingId, () => {
            cleanUp();
            reject();
        });
        link.addEventListener("load", onLoad, {passive: true});
        link.addEventListener("error", onError, {passive: true});
        if (!link.href) {
            onError();
        }
    });
}
function getCSSImportURL(importDeclaration) {
    return getCSSURLValue(
        importDeclaration
            .substring(7)
            .trim()
            .replace(/;$/, "")
            .replace(/screen$/, "")
    );
}
async function loadText(url) {
    if (url.startsWith("data:")) {
        return await (await fetch(url)).text();
    }
    return await bgFetch({
        url,
        responseType: "text",
        mimeType: "text/css",
        origin: window.location.origin
    });
}
async function replaceCSSImports(cssText, basePath, cache = new Map()) {
    cssText = removeCSSComments(cssText);
    cssText = replaceCSSFontFace(cssText);
    cssText = replaceCSSRelativeURLsWithAbsolute(cssText, basePath);
    const importMatches = getMatches(cssImportRegex, cssText);
    for (const match of importMatches) {
        const importURL = getCSSImportURL(match);
        const absoluteURL = getAbsoluteURL(basePath, importURL);
        let importedCSS;
        if (cache.has(absoluteURL)) {
            importedCSS = cache.get(absoluteURL);
        } else {
```

```
            try {
                importedCSS = await loadText(absoluteURL);
                cache.set(absoluteURL, importedCSS);
                importedCSS = await replaceCSSImports(
                    importedCSS,
                    getCSSBaseBath(absoluteURL),
                    cache
                );
            } catch (err) {
                importedCSS = "";
            }
        }
        cssText = cssText.split(match).join(importedCSS);
    }
    cssText = cssText.trim();
    return cssText;
}
function createCORSCopy(srcElement, cssText) {
    if (!cssText) {
        return null;
    }
    const cors = document.createElement("style");
    cors.classList.add("darkreader");
    cors.classList.add("darkreader--cors");
    cors.media = "screen";
    cors.textContent = cssText;
    srcElement.parentNode.insertBefore(cors, srcElement.nextSibling);
    cors.sheet.disabled = true;
    corsStyleSet.add(cors);
    return cors;
}

const observers = [];
let observedRoots;
const definedCustomElements = new Set();
const undefinedGroups = new Map();
let elementsDefinitionCallback;
function isCustomElement(element) {
    if (element.tagName.includes("-") || element.getAttribute("is")) {
        return true;
    }
    return false;
}
function recordUndefinedElement(element) {
    let tag = element.tagName.toLowerCase();
    if (!tag.includes("-")) {
        const extendedTag = element.getAttribute("is");
        if (extendedTag) {
            tag = extendedTag;
        } else {
            return;
        }
    }
    if (!undefinedGroups.has(tag)) {
        undefinedGroups.set(tag, new Set());
        customElementsWhenDefined(tag).then(() => {
            if (elementsDefinitionCallback) {
                const elements = undefinedGroups.get(tag);
                undefinedGroups.delete(tag);
                elementsDefinitionCallback(Array.from(elements));
            }
        });
    }
    undefinedGroups.get(tag).add(element);
}
function collectUndefinedElements(root) {
    if (!isDefinedSelectorSupported) {
        return;
    }
```

```
    forEach(
        root.querySelectorAll(":not(:defined)"),
        recordUndefinedElement
    );
}
let canOptimizeUsingProxy = false;
document.addEventListener(
    "__darkreader__inlineScriptsAllowed",
    () => {
        canOptimizeUsingProxy = true;
    },
    {once: true, passive: true}
);
const resolvers = new Map();
function handleIsDefined(e) {
    canOptimizeUsingProxy = true;
    const tag = e.detail.tag;
    definedCustomElements.add(tag);
    if (resolvers.has(tag)) {
        const r = resolvers.get(tag);
        resolvers.delete(tag);
        r.forEach((r) => r());
    }
}
async function customElementsWhenDefined(tag) {
    if (definedCustomElements.has(tag)) {
        return;
    }
    return new Promise((resolve) => {
        if (
            window.customElements &&
            typeof customElements.whenDefined === "function"
        ) {
            customElements.whenDefined(tag).then(() => resolve());
        } else if (canOptimizeUsingProxy) {
            if (resolvers.has(tag)) {
                resolvers.get(tag).push(resolve);
            } else {
                resolvers.set(tag, [resolve]);
            }
            document.dispatchEvent(
                new CustomEvent("__darkreader__addUndefinedResolver", {
                    detail: {tag}
                })
            );
        } else {
            const checkIfDefined = () => {
                const elements = undefinedGroups.get(tag);
                if (elements && elements.size > 0) {
                    if (
                        elements.values().next().value.matches(":defined")
                    ) {
                        resolve();
                    } else {
                        requestAnimationFrame(checkIfDefined);
                    }
                }
            };
            requestAnimationFrame(checkIfDefined);
        }
    });
}
function watchWhenCustomElementsDefined(callback) {
    elementsDefinitionCallback = callback;
}
function unsubscribeFromDefineCustomElements() {
    elementsDefinitionCallback = null;
    undefinedGroups.clear();
    document.removeEventListener(
```

```
        "__darkreader__isDefined",
        handleIsDefined
    );
}
function watchForStyleChanges(currentStyles, update, shadowRootDiscovered) {
    stopWatchingForStyleChanges();
    const prevStyles = new Set(currentStyles);
    const prevStyleSiblings = new WeakMap();
    const nextStyleSiblings = new WeakMap();
    function saveStylePosition(style) {
        prevStyleSiblings.set(style, style.previousElementSibling);
        nextStyleSiblings.set(style, style.nextElementSibling);
    }
    function forgetStylePosition(style) {
        prevStyleSiblings.delete(style);
        nextStyleSiblings.delete(style);
    }
    function didStylePositionChange(style) {
        return (
            style.previousElementSibling !== prevStyleSiblings.get(style) ||
            style.nextElementSibling !== nextStyleSiblings.get(style)
        );
    }
    currentStyles.forEach(saveStylePosition);
    function handleStyleOperations(operations) {
        const {createdStyles, removedStyles, movedStyles} = operations;
        createdStyles.forEach((s) => saveStylePosition(s));
        movedStyles.forEach((s) => saveStylePosition(s));
        removedStyles.forEach((s) => forgetStylePosition(s));
        createdStyles.forEach((s) => prevStyles.add(s));
        removedStyles.forEach((s) => prevStyles.delete(s));
        if (
            createdStyles.size + removedStyles.size + movedStyles.size >
            0
        ) {
            update({
                created: Array.from(createdStyles),
                removed: Array.from(removedStyles),
                moved: Array.from(movedStyles),
                updated: []
            });
        }
    }
    function handleMinorTreeMutations({additions, moves, deletions}) {
        const createdStyles = new Set();
        const removedStyles = new Set();
        const movedStyles = new Set();
        additions.forEach((node) =>
            getManageableStyles(node).forEach((style) =>
                createdStyles.add(style)
            )
        );
        deletions.forEach((node) =>
            getManageableStyles(node).forEach((style) =>
                removedStyles.add(style)
            )
        );
        moves.forEach((node) =>
            getManageableStyles(node).forEach((style) =>
                movedStyles.add(style)
            )
        );
        handleStyleOperations({createdStyles, removedStyles, movedStyles});
        additions.forEach((n) => {
            extendedIterateShadowHosts(n);
            collectUndefinedElements(n);
        });
        additions.forEach(
            (node) => isCustomElement(node) && recordUndefinedElement(node)
```

```
    );
}
function handleHugeTreeMutations(root) {
    const styles = new Set(getManageableStyles(root));
    const createdStyles = new Set();
    const removedStyles = new Set();
    const movedStyles = new Set();
    styles.forEach((s) => {
        if (!prevStyles.has(s)) {
            createdStyles.add(s);
        }
    });
    prevStyles.forEach((s) => {
        if (!styles.has(s)) {
            removedStyles.add(s);
        }
    });
    styles.forEach((s) => {
        if (
            !createdStyles.has(s) &&
            !removedStyles.has(s) &&
            didStylePositionChange(s)
        ) {
            movedStyles.add(s);
        }
    });
    handleStyleOperations({createdStyles, removedStyles, movedStyles});
    extendedIterateShadowHosts(root);
    collectUndefinedElements(root);
}
function handleAttributeMutations(mutations) {
    const updatedStyles = new Set();
    const removedStyles = new Set();
    mutations.forEach((m) => {
        const {target} = m;
        if (target.isConnected) {
            if (shouldManageStyle(target)) {
                updatedStyles.add(target);
            } else if (
                target instanceof HTMLLinkElement &&
                target.disabled
            ) {
                removedStyles.add(target);
            }
        }
    });
    if (updatedStyles.size + removedStyles.size > 0) {
        update({
            updated: Array.from(updatedStyles),
            created: [],
            removed: Array.from(removedStyles),
            moved: []
        });
    }
}
function observe(root) {
    if (observedRoots.has(root)) {
        return;
    }
    const treeObserver = createOptimizedTreeObserver(root, {
        onMinorMutations: handleMinorTreeMutations,
        onHugeMutations: handleHugeTreeMutations
    });
    const attrObserver = new MutationObserver(handleAttributeMutations);
    attrObserver.observe(root, {
        attributeFilter: ["rel", "disabled", "media", "href"],
        subtree: true
    });
    observers.push(treeObserver, attrObserver);
```

```
        observedRoots.add(root);
    }
    function subscribeForShadowRootChanges(node) {
        const {shadowRoot} = node;
        if (shadowRoot == null || observedRoots.has(shadowRoot)) {
            return;
        }
        observe(shadowRoot);
        shadowRootDiscovered(shadowRoot);
    }
    function extendedIterateShadowHosts(node) {
        iterateShadowHosts(node, subscribeForShadowRootChanges);
    }
    observe(document);
    extendedIterateShadowHosts(document.documentElement);
    watchWhenCustomElementsDefined((hosts) => {
        const newStyles = [];
        hosts.forEach((host) =>
            push(newStyles, getManageableStyles(host.shadowRoot))
        );
        update({created: newStyles, updated: [], removed: [], moved: []});
        hosts.forEach((host) => {
            const {shadowRoot} = host;
            if (shadowRoot == null) {
                return;
            }
            subscribeForShadowRootChanges(host);
            extendedIterateShadowHosts(shadowRoot);
            collectUndefinedElements(shadowRoot);
        });
    });
    document.addEventListener("__darkreader__isDefined", handleIsDefined, {
        passive: true
    });
    collectUndefinedElements(document);
}
function resetObservers() {
    observers.forEach((o) => o.disconnect());
    observers.splice(0, observers.length);
    observedRoots = new WeakSet();
}
function stopWatchingForStyleChanges() {
    resetObservers();
    unsubscribeFromDefineCustomElements();
}

const overrides = new WeakSet();
function hasAdoptedStyleSheets(node) {
    return (
        Array.isArray(node.adoptedStyleSheets) &&
        node.adoptedStyleSheets.length > 0
    );
}
function createAdoptedStyleSheetOverride(node) {
    let cancelAsyncOperations = false;
    function iterateSourceSheets(iterator) {
        node.adoptedStyleSheets.forEach((sheet) => {
            if (!overrides.has(sheet)) {
                iterator(sheet);
            }
        });
    }
    function injectSheet(sheet, override) {
        const newSheets = [...node.adoptedStyleSheets];
        const sheetIndex = newSheets.indexOf(sheet);
        const overrideIndex = newSheets.indexOf(override);
        if (overrideIndex >= 0) {
            newSheets.splice(overrideIndex, 1);
        }
```

```
    newSheets.splice(sheetIndex + 1, 0, override);
    node.adoptedStyleSheets = newSheets;
}
function clear() {
    const newSheets = [...node.adoptedStyleSheets];
    for (let i = newSheets.length - 1; i >= 0; i--) {
        const sheet = newSheets[i];
        if (overrides.has(sheet)) {
            newSheets.splice(i, 1);
            overrides.delete(sheet);
        }
    }
    if (node.adoptedStyleSheets.length !== newSheets.length) {
        node.adoptedStyleSheets = newSheets;
    }
}
function destroy() {
    cancelAsyncOperations = true;
    clear();
    if (frameId) {
        cancelAnimationFrame(frameId);
        frameId = null;
    }
}
let rulesChangeKey = 0;
function getRulesChangeKey() {
    let count = 0;
    iterateSourceSheets((sheet) => {
        count += sheet.cssRules.length;
    });
    if (count === 1) {
        const rule = node.adoptedStyleSheets[0].cssRules[0];
        return rule instanceof CSSStyleRule ? rule.style.length : count;
    }
    return count;
}
function render(theme, ignoreImageAnalysis) {
    clear();
    for (let i = node.adoptedStyleSheets.length - 1; i >= 0; i--) {
        const sheet = node.adoptedStyleSheets[i];
        if (overrides.has(sheet)) {
            continue;
        }
        const rules = sheet.cssRules;
        const override = new CSSStyleSheet();
        const prepareSheet = () => {
            for (let i = override.cssRules.length - 1; i >= 0; i--) {
                override.deleteRule(i);
            }
            injectSheet(sheet, override);
            overrides.add(override);
            return override;
        };
        const sheetModifier = createStyleSheetModifier();
        sheetModifier.modifySheet({
            prepareSheet,
            sourceCSSRules: rules,
            theme,
            ignoreImageAnalysis,
            force: false,
            isAsyncCancelled: () => cancelAsyncOperations
        });
    }
    rulesChangeKey = getRulesChangeKey();
}
function checkForUpdates() {
    return getRulesChangeKey() !== rulesChangeKey;
}
let frameId = null;
```

```
    function watch(callback) {
        frameId = requestAnimationFrame(() => {
            if (checkForUpdates()) {
                const sheets = node.adoptedStyleSheets.filter(
                    (s) => !overrides.has(s)
                );
                callback(sheets);
            }
            watch(callback);
        });
    }
    return {
        render,
        destroy,
        watch
    };
}

function injectProxy(
    enableStyleSheetsProxy,
    enableCustomElementRegistryProxy
) {
    document.dispatchEvent(
        new CustomEvent("__darkreader__inlineScriptsAllowed")
    );
    const addRuleDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "addRule"
    );
    const insertRuleDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "insertRule"
    );
    const deleteRuleDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "deleteRule"
    );
    const removeRuleDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "removeRule"
    );
    const replaceDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "replace"
    );
    const replaceSyncDescriptor = Object.getOwnPropertyDescriptor(
        CSSStyleSheet.prototype,
        "replaceSync"
    );
    const documentStyleSheetsDescriptor = enableStyleSheetsProxy
        ? Object.getOwnPropertyDescriptor(Document.prototype, "styleSheets")
        : null;
    const customElementRegistryDefineDescriptor =
        enableCustomElementRegistryProxy
            ? Object.getOwnPropertyDescriptor(
                  CustomElementRegistry.prototype,
                  "define"
              )
            : null;
    const shouldWrapHTMLElement = [
        "baidu.com",
        "baike.baidu.com",
        "ditu.baidu.com",
        "map.baidu.com",
        "maps.baidu.com",
        "haokan.baidu.com",
        "pan.baidu.com",
        "passport.baidu.com",
        "tieba.baidu.com",
```

```
    "www.baidu.com"
].includes(location.hostname);
const getElementsByTagNameDescriptor = shouldWrapHTMLElement
    ? Object.getOwnPropertyDescriptor(
          Element.prototype,
          "getElementsByTagName"
      )
    : null;
const shouldProxyChildNodes = location.hostname === "www.vy.no";
const childNodesDescriptor = shouldProxyChildNodes
    ? Object.getOwnPropertyDescriptor(Node.prototype, "childNodes")
    : null;
const cleaners = [];
const cleanUp = () => {
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "addRule",
        addRuleDescriptor
    );
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "insertRule",
        insertRuleDescriptor
    );
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "deleteRule",
        deleteRuleDescriptor
    );
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "removeRule",
        removeRuleDescriptor
    );
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "replace",
        replaceDescriptor
    );
    Object.defineProperty(
        CSSStyleSheet.prototype,
        "replaceSync",
        replaceSyncDescriptor
    );
    document.removeEventListener("__darkreader__cleanUp", cleanUp);
    document.removeEventListener(
        "__darkreader__addUndefinedResolver",
        addUndefinedResolver
    );
    document.removeEventListener(
        "__darkreader__blobURLCheckRequest",
        checkBlobURLSupport
    );
    if (enableStyleSheetsProxy) {
        Object.defineProperty(
            Document.prototype,
            "styleSheets",
            documentStyleSheetsDescriptor
        );
    }
    if (enableCustomElementRegistryProxy) {
        Object.defineProperty(
            CustomElementRegistry.prototype,
            "define",
            customElementRegistryDefineDescriptor
        );
    }
    if (shouldWrapHTMLElement) {
        Object.defineProperty(
```

```
            Element.prototype,
            "getElementsByTagName",
            getElementsByTagNameDescriptor
        );
    }
    if (shouldProxyChildNodes) {
        Object.defineProperty(
            Node.prototype,
            "childNodes",
            childNodesDescriptor
        );
    }
    cleaners.forEach((clean) => clean());
    cleaners.splice(0);
};
const addUndefinedResolverInner = (tag) => {
    customElements.whenDefined(tag).then(() => {
        document.dispatchEvent(
            new CustomEvent("__darkreader__isDefined", {detail: {tag}})
        );
    });
};
const addUndefinedResolver = (e) =>
    addUndefinedResolverInner(e.detail.tag);
document.addEventListener("__darkreader__cleanUp", cleanUp, {
    passive: true
});
document.addEventListener(
    "__darkreader__addUndefinedResolver",
    addUndefinedResolver,
    {passive: true}
);
document.addEventListener(
    "__darkreader__blobURLCheckRequest",
    checkBlobURLSupport,
    {once: true}
);
const updateSheetEvent = new Event("__darkreader__updateSheet");
function proxyAddRule(selector, style, index) {
    addRuleDescriptor.value.call(this, selector, style, index);
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
    return -1;
}
function proxyInsertRule(rule, index) {
    const returnValue = insertRuleDescriptor.value.call(
        this,
        rule,
        index
    );
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
    return returnValue;
}
function proxyDeleteRule(index) {
```

```
    deleteRuleDescriptor.value.call(this, index);
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
}
function proxyRemoveRule(index) {
    removeRuleDescriptor.value.call(this, index);
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
}
function proxyReplace(cssText) {
    const returnValue = replaceDescriptor.value.call(this, cssText);
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        ) &&
        returnValue &&
        returnValue instanceof Promise
    ) {
        returnValue.then(() =>
            this.ownerNode.dispatchEvent(updateSheetEvent)
        );
    }
    return returnValue;
}
function proxyReplaceSync(cssText) {
    replaceSyncDescriptor.value.call(this, cssText);
    if (
        this.ownerNode &&
        !(
            this.ownerNode.classList &&
            this.ownerNode.classList.contains("darkreader")
        )
    ) {
        this.ownerNode.dispatchEvent(updateSheetEvent);
    }
}
function proxyDocumentStyleSheets() {
    const getCurrentValue = () => {
        const docSheets = documentStyleSheetsDescriptor.get.call(this);
        const filteredSheets = [...docSheets].filter(
            (styleSheet) =>
                styleSheet.ownerNode &&
                !(
                    styleSheet.ownerNode.classList &&
                    styleSheet.ownerNode.classList.contains(
                        "darkreader"
                    )
                )
        );
        filteredSheets.item = (item) => filteredSheets[item];
        return Object.setPrototypeOf(
            filteredSheets,
            StyleSheetList.prototype
```

```
            );
        };
        let elements = getCurrentValue();
        const styleSheetListBehavior = {
            get: function (_, property) {
                return getCurrentValue()[property];
            }
        };
        elements = new Proxy(elements, styleSheetListBehavior);
        return elements;
    }
    function proxyCustomElementRegistryDefine(name, constructor, options) {
        addUndefinedResolverInner(name);
        customElementRegistryDefineDescriptor.value.call(
            this,
            name,
            constructor,
            options
        );
    }
    function proxyGetElementsByTagName(tagName) {
        if (tagName !== "style") {
            return getElementsByTagNameDescriptor.value.call(this, tagName);
        }
        const getCurrentElementValue = () => {
            const elements = getElementsByTagNameDescriptor.value.call(
                this,
                tagName
            );
            return Object.setPrototypeOf(
                [...elements].filter(
                    (element) =>
                        element &&
                        !(
                            element.classList &&
                            element.classList.contains("darkreader")
                        )
                ),
                NodeList.prototype
            );
        };
        let elements = getCurrentElementValue();
        const nodeListBehavior = {
            get: function (_, property) {
                return getCurrentElementValue()[
                    Number(property) || property
                ];
            }
        };
        elements = new Proxy(elements, nodeListBehavior);
        return elements;
    }
    function proxyChildNodes() {
        const childNodes = childNodesDescriptor.get.call(this);
        return Object.setPrototypeOf(
            [...childNodes].filter((element) => {
                return (
                    !element.classList ||
                    !element.classList.contains("darkreader")
                );
            }),
            NodeList.prototype
        );
    }
    async function checkBlobURLSupport() {
        const svg =
            '<svg xmlns="http://www.w3.org/2000/svg" width="1" height="1"><rect width="1" height="1"
fill="transparent"/></svg>';
        const bytes = new Uint8Array(svg.length);
```

```
    for (let i = 0; i < svg.length; i++) {
        bytes[i] = svg.charCodeAt(i);
    }
    const blob = new Blob([bytes], {type: "image/svg+xml"});
    const objectURL = URL.createObjectURL(blob);
    let blobURLAllowed;
    try {
        const image = new Image();
        await new Promise((resolve, reject) => {
            image.onload = () => resolve();
            image.onerror = () => reject();
            image.src = objectURL;
        });
        blobURLAllowed = true;
    } catch (err) {
        blobURLAllowed = false;
    }
    document.dispatchEvent(
        new CustomEvent("__darkreader__blobURLCheckResponse", {
            detail: {blobURLAllowed}
        })
    );
}
Object.defineProperty(CSSStyleSheet.prototype, "addRule", {
    ...addRuleDescriptor,
    value: proxyAddRule
});
Object.defineProperty(CSSStyleSheet.prototype, "insertRule", {
    ...insertRuleDescriptor,
    value: proxyInsertRule
});
Object.defineProperty(CSSStyleSheet.prototype, "deleteRule", {
    ...deleteRuleDescriptor,
    value: proxyDeleteRule
});
Object.defineProperty(CSSStyleSheet.prototype, "removeRule", {
    ...removeRuleDescriptor,
    value: proxyRemoveRule
});
Object.defineProperty(CSSStyleSheet.prototype, "replace", {
    ...replaceDescriptor,
    value: proxyReplace
});
Object.defineProperty(CSSStyleSheet.prototype, "replaceSync", {
    ...replaceSyncDescriptor,
    value: proxyReplaceSync
});
if (enableStyleSheetsProxy) {
    Object.defineProperty(Document.prototype, "styleSheets", {
        ...documentStyleSheetsDescriptor,
        get: proxyDocumentStyleSheets
    });
}
if (enableCustomElementRegistryProxy) {
    Object.defineProperty(CustomElementRegistry.prototype, "define", {
        ...customElementRegistryDefineDescriptor,
        value: proxyCustomElementRegistryDefine
    });
}
if (shouldWrapHTMLElement) {
    Object.defineProperty(Element.prototype, "getElementsByTagName", {
        ...getElementsByTagNameDescriptor,
        value: proxyGetElementsByTagName
    });
}
if (shouldProxyChildNodes) {
    Object.defineProperty(Node.prototype, "childNodes", {
        ...childNodesDescriptor,
        get: proxyChildNodes
```

```
        });
    }
}

let documentVisibilityListener = null;
let documentIsVisible_ = !document.hidden;
const listenerOptions = {
    capture: true,
    passive: true
};
function watchForDocumentVisibility() {
    document.addEventListener(
        "visibilitychange",
        documentVisibilityListener,
        listenerOptions
    );
    window.addEventListener(
        "pageshow",
        documentVisibilityListener,
        listenerOptions
    );
    window.addEventListener(
        "focus",
        documentVisibilityListener,
        listenerOptions
    );
}
function stopWatchingForDocumentVisibility() {
    document.removeEventListener(
        "visibilitychange",
        documentVisibilityListener,
        listenerOptions
    );
    window.removeEventListener(
        "pageshow",
        documentVisibilityListener,
        listenerOptions
    );
    window.removeEventListener(
        "focus",
        documentVisibilityListener,
        listenerOptions
    );
}
function setDocumentVisibilityListener(callback) {
    const alreadyWatching = Boolean(documentVisibilityListener);
    documentVisibilityListener = () => {
        if (!document.hidden) {
            removeDocumentVisibilityListener();
            callback();
            documentIsVisible_ = true;
        }
    };
    if (!alreadyWatching) {
        watchForDocumentVisibility();
    }
}
function removeDocumentVisibilityListener() {
    stopWatchingForDocumentVisibility();
    documentVisibilityListener = null;
}
function documentIsVisible() {
    return documentIsVisible_;
}

function findRelevantFix(documentURL, fixes) {
    if (
        !Array.isArray(fixes) ||
        fixes.length === 0 ||
```

```
        fixes[0].url[0] !== "*"
    ) {
        return null;
    }
    let maxSpecificity = 0;
    let maxSpecificityIndex = null;
    for (let i = 1; i < fixes.length; i++) {
        if (isURLInList(documentURL, fixes[i].url)) {
            const specificity = fixes[i].url[0].length;
            if (
                maxSpecificityIndex === null ||
                maxSpecificity < specificity
            ) {
                maxSpecificity = specificity;
                maxSpecificityIndex = i;
            }
        }
    }
    return maxSpecificityIndex;
}
function combineFixes(fixes) {
    if (fixes.length === 0 || fixes[0].url[0] !== "*") {
        return null;
    }
    function combineArrays(arrays) {
        return arrays.filter(Boolean).flat();
    }
    return {
        url: [],
        invert: combineArrays(fixes.map((fix) => fix.invert)),
        css: fixes
            .map((fix) => fix.css)
            .filter(Boolean)
            .join("\n"),
        ignoreInlineStyle: combineArrays(
            fixes.map((fix) => fix.ignoreInlineStyle)
        ),
        ignoreImageAnalysis: combineArrays(
            fixes.map((fix) => fix.ignoreImageAnalysis)
        ),
        disableStyleSheetsProxy: fixes.some(
            (fix) => fix.disableStyleSheetsProxy
        ),
        disableCustomElementRegistryProxy: fixes.some(
            (fix) => fix.disableCustomElementRegistryProxy
        )
    };
}

const INSTANCE_ID = generateUID();
const styleManagers = new Map();
const adoptedStyleManagers = [];
const adoptedStyleFallbacks = new Map();
let filter = null;
let fixes = null;
let isIFrame = null;
let ignoredImageAnalysisSelectors = [];
let ignoredInlineSelectors = [];
function createOrUpdateStyle(className, root = document.head || document) {
    let element = root.querySelector(`.${className}`);
    if (!element) {
        element = document.createElement("style");
        element.classList.add("darkreader");
        element.classList.add(className);
        element.media = "screen";
        element.textContent = "";
    }
    return element;
}
```

```
function createOrUpdateScript(className, root = document.head || document) {
    let element = root.querySelector(`.${className}`);
    if (!element) {
        element = document.createElement("script");
        element.classList.add("darkreader");
        element.classList.add(className);
    }
    return element;
}
const nodePositionWatchers = new Map();
function setupNodePositionWatcher(node, alias) {
    nodePositionWatchers.has(alias) &&
        nodePositionWatchers.get(alias).stop();
    nodePositionWatchers.set(alias, watchForNodePosition(node, "head"));
}
function stopStylePositionWatchers() {
    forEach(nodePositionWatchers.values(), (watcher) => watcher.stop());
    nodePositionWatchers.clear();
}
function createStaticStyleOverrides() {
    const fallbackStyle = createOrUpdateStyle(
        "darkreader--fallback",
        document
    );
    fallbackStyle.textContent = getModifiedFallbackStyle(filter, {
        strict: true
    });
    document.head.insertBefore(fallbackStyle, document.head.firstChild);
    setupNodePositionWatcher(fallbackStyle, "fallback");
    const userAgentStyle = createOrUpdateStyle("darkreader--user-agent");
    userAgentStyle.textContent = getModifiedUserAgentStyle(
        filter,
        isIFrame,
        filter.styleSystemControls
    );
    document.head.insertBefore(userAgentStyle, fallbackStyle.nextSibling);
    setupNodePositionWatcher(userAgentStyle, "user-agent");
    const textStyle = createOrUpdateStyle("darkreader--text");
    if (filter.useFont || filter.textStroke > 0) {
        textStyle.textContent = createTextStyle(filter);
    } else {
        textStyle.textContent = "";
    }
    document.head.insertBefore(textStyle, fallbackStyle.nextSibling);
    setupNodePositionWatcher(textStyle, "text");
    const invertStyle = createOrUpdateStyle("darkreader--invert");
    if (fixes && Array.isArray(fixes.invert) && fixes.invert.length > 0) {
        invertStyle.textContent = [
            `${fixes.invert.join(", ")} {`,
            `    filter: ${getCSSFilterValue({
                ...filter,
                contrast:
                    filter.mode === 0
                        ? filter.contrast
                        : clamp(filter.contrast - 10, 0, 100)
            })} !important;`,
            "}"
        ].join("\n");
    } else {
        invertStyle.textContent = "";
    }
    document.head.insertBefore(invertStyle, textStyle.nextSibling);
    setupNodePositionWatcher(invertStyle, "invert");
    const inlineStyle = createOrUpdateStyle("darkreader--inline");
    inlineStyle.textContent = getInlineOverrideStyle();
    document.head.insertBefore(inlineStyle, invertStyle.nextSibling);
    setupNodePositionWatcher(inlineStyle, "inline");
    const overrideStyle = createOrUpdateStyle("darkreader--override");
    overrideStyle.textContent =
```

```
        fixes && fixes.css ? replaceCSSTemplates(fixes.css) : "";
    document.head.appendChild(overrideStyle);
    setupNodePositionWatcher(overrideStyle, "override");
    const variableStyle = createOrUpdateStyle("darkreader--variables");
    const selectionColors = getSelectionColor(filter);
    const {
        darkSchemeBackgroundColor,
        darkSchemeTextColor,
        lightSchemeBackgroundColor,
        lightSchemeTextColor,
        mode
    } = filter;
    let schemeBackgroundColor =
        mode === 0 ? lightSchemeBackgroundColor : darkSchemeBackgroundColor;
    let schemeTextColor =
        mode === 0 ? lightSchemeTextColor : darkSchemeTextColor;
    schemeBackgroundColor = modifyBackgroundColor(
        parseColorWithCache(schemeBackgroundColor),
        filter
    );
    schemeTextColor = modifyForegroundColor(
        parseColorWithCache(schemeTextColor),
        filter
    );
    variableStyle.textContent = [
        `:root {`,
        `   --darkreader-neutral-background: ${schemeBackgroundColor};`,
        `   --darkreader-neutral-text: ${schemeTextColor};`,
        `   --darkreader-selection-background: ${selectionColors.backgroundColorSelection};`,
        `   --darkreader-selection-text: ${selectionColors.foregroundColorSelection};`,
        `}`
    ].join("\n");
    document.head.insertBefore(variableStyle, inlineStyle.nextSibling);
    setupNodePositionWatcher(variableStyle, "variables");
    const rootVarsStyle = createOrUpdateStyle("darkreader--root-vars");
    document.head.insertBefore(rootVarsStyle, variableStyle.nextSibling);
    const enableStyleSheetsProxy = !(
        fixes && fixes.disableStyleSheetsProxy
    );
    const enableCustomElementRegistryProxy = !(
        fixes && fixes.disableCustomElementRegistryProxy
    );
    {
        const proxyScript = createOrUpdateScript("darkreader--proxy");
        proxyScript.append(
            `(${injectProxy})(${enableStyleSheetsProxy}, ${enableCustomElementRegistryProxy})`
        );
        document.head.insertBefore(proxyScript, rootVarsStyle.nextSibling);
        proxyScript.remove();
    }
}
const shadowRootsWithOverrides = new Set();
function createShadowStaticStyleOverridesInner(root) {
    const inlineStyle = createOrUpdateStyle("darkreader--inline", root);
    inlineStyle.textContent = getInlineOverrideStyle();
    root.insertBefore(inlineStyle, root.firstChild);
    const overrideStyle = createOrUpdateStyle("darkreader--override", root);
    overrideStyle.textContent =
        fixes && fixes.css ? replaceCSSTemplates(fixes.css) : "";
    root.insertBefore(overrideStyle, inlineStyle.nextSibling);
    const invertStyle = createOrUpdateStyle("darkreader--invert", root);
    if (fixes && Array.isArray(fixes.invert) && fixes.invert.length > 0) {
        invertStyle.textContent = [
            `${fixes.invert.join(", ")} {`,
            `    filter: ${getCSSFilterValue({
                ...filter,
                contrast:
                    filter.mode === 0
                        ? filter.contrast
```

```
                        : clamp(filter.contrast - 10, 0, 100)
            })} !important;`,
            "}"
        ].join("\n");
    } else {
        invertStyle.textContent = "";
    }
    root.insertBefore(invertStyle, overrideStyle.nextSibling);
    shadowRootsWithOverrides.add(root);
}
function delayedCreateShadowStaticStyleOverrides(root) {
    const observer = new MutationObserver((mutations, observer) => {
        observer.disconnect();
        for (const {type, removedNodes} of mutations) {
            if (type === "childList") {
                for (const {nodeName, className} of removedNodes) {
                    if (
                        nodeName === "STYLE" &&
                        [
                            "darkreader darkreader--inline",
                            "darkreader darkreader--override",
                            "darkreader darkreader--invert"
                        ].includes(className)
                    ) {
                        createShadowStaticStyleOverridesInner(root);
                        return;
                    }
                }
            }
        }
    });
    observer.observe(root, {childList: true});
}
function createShadowStaticStyleOverrides(root) {
    const uninit = root.firstChild === null;
    createShadowStaticStyleOverridesInner(root);
    if (uninit) {
        delayedCreateShadowStaticStyleOverrides(root);
    }
}
function replaceCSSTemplates($cssText) {
    return $cssText.replace(/\${(.+?)}/g, (_, $color) => {
        const color = parseColorWithCache($color);
        if (color) {
            return modifyColor(color, filter);
        }
        return $color;
    });
}
function cleanFallbackStyle() {
    const fallback = document.querySelector(".darkreader--fallback");
    if (fallback) {
        fallback.textContent = "";
    }
}
function createDynamicStyleOverrides() {
    cancelRendering();
    const allStyles = getManageableStyles(document);
    const newManagers = allStyles
        .filter((style) => !styleManagers.has(style))
        .map((style) => createManager(style));
    newManagers
        .map((manager) => manager.details({secondRound: false}))
        .filter((detail) => detail && detail.rules.length > 0)
        .forEach((detail) => {
            variablesStore.addRulesForMatching(detail.rules);
        });
    variablesStore.matchVariablesAndDependents();
    variablesStore.setOnRootVariableChange(() => {
```

```
        const rootVarsStyle = createOrUpdateStyle("darkreader--root-vars");
        variablesStore.putRootVars(rootVarsStyle, filter);
    });
    const rootVarsStyle = createOrUpdateStyle("darkreader--root-vars");
    variablesStore.putRootVars(rootVarsStyle, filter);
    styleManagers.forEach((manager) =>
        manager.render(filter, ignoredImageAnalysisSelectors)
    );
    if (loadingStyles.size === 0) {
        cleanFallbackStyle();
    }
    newManagers.forEach((manager) => manager.watch());
    const inlineStyleElements = toArray(
        document.querySelectorAll(INLINE_STYLE_SELECTOR)
    );
    iterateShadowHosts(document.documentElement, (host) => {
        createShadowStaticStyleOverrides(host.shadowRoot);
        const elements = host.shadowRoot.querySelectorAll(
            INLINE_STYLE_SELECTOR
        );
        if (elements.length > 0) {
            push(inlineStyleElements, elements);
        }
    });
    inlineStyleElements.forEach((el) =>
        overrideInlineStyle(
            el,
            filter,
            ignoredInlineSelectors,
            ignoredImageAnalysisSelectors
        )
    );
    handleAdoptedStyleSheets(document);
    variablesStore.matchVariablesAndDependents();
}
let loadingStylesCounter = 0;
const loadingStyles = new Set();
function createManager(element) {
    const loadingStyleId = ++loadingStylesCounter;
    function loadingStart() {
        if (!isDOMReady() || !documentIsVisible()) {
            loadingStyles.add(loadingStyleId);
            logInfo(
                `Current amount of styles loading: ${loadingStyles.size}`
            );
            const fallbackStyle = document.querySelector(
                ".darkreader--fallback"
            );
            if (!fallbackStyle.textContent) {
                fallbackStyle.textContent = getModifiedFallbackStyle(
                    filter,
                    {strict: false}
                );
            }
        }
    }
    function loadingEnd() {
        loadingStyles.delete(loadingStyleId);
        logInfo(
            `Removed loadingStyle ${loadingStyleId}, now awaiting: ${loadingStyles.size}`
        );
        if (loadingStyles.size === 0 && isDOMReady()) {
            cleanFallbackStyle();
        }
    }
    function update() {
        const details = manager.details({secondRound: true});
        if (!details) {
            return;
```

```
        }
        variablesStore.addRulesForMatching(details.rules);
        variablesStore.matchVariablesAndDependents();
        manager.render(filter, ignoredImageAnalysisSelectors);
    }
    const manager = manageStyle(element, {
        update,
        loadingStart,
        loadingEnd
    });
    styleManagers.set(element, manager);
    return manager;
}
function removeManager(element) {
    const manager = styleManagers.get(element);
    if (manager) {
        manager.destroy();
        styleManagers.delete(element);
    }
}
const throttledRenderAllStyles = throttle((callback) => {
    styleManagers.forEach((manager) =>
        manager.render(filter, ignoredImageAnalysisSelectors)
    );
    adoptedStyleManagers.forEach((manager) =>
        manager.render(filter, ignoredImageAnalysisSelectors)
    );
    callback && callback();
});
const cancelRendering = function () {
    throttledRenderAllStyles.cancel();
};
function onDOMReady() {
    if (loadingStyles.size === 0) {
        cleanFallbackStyle();
        return;
    }
}
function runDynamicStyle() {
    createDynamicStyleOverrides();
    watchForUpdates();
}
function createThemeAndWatchForUpdates() {
    createStaticStyleOverrides();
    if (!documentIsVisible() && !filter.immediateModify) {
        setDocumentVisibilityListener(runDynamicStyle);
    } else {
        runDynamicStyle();
    }
    changeMetaThemeColorWhenAvailable(filter);
}
let pendingAdoptedVarMatch = false;
function handleAdoptedStyleSheets(node) {
    const theme = filter;
    if (hasAdoptedStyleSheets(node)) {
        node.adoptedStyleSheets.forEach((s) => {
            variablesStore.addRulesForMatching(s.cssRules);
        });
        const newManger = createAdoptedStyleSheetOverride(node);
        adoptedStyleManagers.push(newManger);
        newManger.render(theme, ignoredImageAnalysisSelectors);
        newManger.watch((sheets) => {
            sheets.forEach((s) => {
                variablesStore.addRulesForMatching(s.cssRules);
            });
            newManger.render(theme, ignoredImageAnalysisSelectors);
            pendingAdoptedVarMatch = true;
        });
        potentialAdoptedStyleNodes.delete(node);
```

```
    } else if (!potentialAdoptedStyleNodes.has(node)) {
        potentialAdoptedStyleNodes.add(node);
    }
}
const potentialAdoptedStyleNodes = new Set();
let potentialAdoptedStyleFrameId = null;
function watchPotentialAdoptedStyleNodes() {
    potentialAdoptedStyleFrameId = requestAnimationFrame(() => {
        let changed = false;
        potentialAdoptedStyleNodes.forEach((node) => {
            if (node.isConnected) {
                handleAdoptedStyleSheets(node);
                changed = true;
            } else {
                potentialAdoptedStyleNodes.delete(node);
            }
        });
        if (changed || pendingAdoptedVarMatch) {
            variablesStore.matchVariablesAndDependents();
            pendingAdoptedVarMatch = false;
        }
        watchPotentialAdoptedStyleNodes();
    });
}
function stopWatchingPotentialAdoptedStyleNodes() {
    potentialAdoptedStyleFrameId &&
        cancelAnimationFrame(potentialAdoptedStyleFrameId);
    potentialAdoptedStyleNodes.clear();
}
function watchForUpdates() {
    const managedStyles = Array.from(styleManagers.keys());
    watchForStyleChanges(
        managedStyles,
        ({created, updated, removed, moved}) => {
            const stylesToRemove = removed;
            const stylesToManage = created
                .concat(updated)
                .concat(moved)
                .filter((style) => !styleManagers.has(style));
            const stylesToRestore = moved.filter((style) =>
                styleManagers.has(style)
            );
            stylesToRemove.forEach((style) => removeManager(style));
            const newManagers = stylesToManage.map((style) =>
                createManager(style)
            );
            newManagers
                .map((manager) => manager.details({secondRound: false}))
                .filter((detail) => detail && detail.rules.length > 0)
                .forEach((detail) => {
                    variablesStore.addRulesForMatching(detail.rules);
                });
            variablesStore.matchVariablesAndDependents();
            newManagers.forEach((manager) =>
                manager.render(filter, ignoredImageAnalysisSelectors)
            );
            newManagers.forEach((manager) => manager.watch());
            stylesToRestore.forEach((style) =>
                styleManagers.get(style).restore()
            );
        },
        (shadowRoot) => {
            createShadowStaticStyleOverrides(shadowRoot);
            handleAdoptedStyleSheets(shadowRoot);
        }
    );
    watchPotentialAdoptedStyleNodes();
    watchForInlineStyles(
        (element) => {
```

```
            overrideInlineStyle(
                element,
                filter,
                ignoredInlineSelectors,
                ignoredImageAnalysisSelectors
            );
            if (element === document.documentElement) {
                const styleAttr = element.getAttribute("style") || "";
                if (styleAttr.includes("--")) {
                    variablesStore.matchVariablesAndDependents();
                    const rootVarsStyle = createOrUpdateStyle(
                        "darkreader--root-vars"
                    );
                    variablesStore.putRootVars(rootVarsStyle, filter);
                }
            }
        },
        (root) => {
            createShadowStaticStyleOverrides(root);
            const inlineStyleElements = root.querySelectorAll(
                INLINE_STYLE_SELECTOR
            );
            if (inlineStyleElements.length > 0) {
                forEach(inlineStyleElements, (el) =>
                    overrideInlineStyle(
                        el,
                        filter,
                        ignoredInlineSelectors,
                        ignoredImageAnalysisSelectors
                    )
                );
            }
        }
    );
    addDOMReadyListener(onDOMReady);
}
function stopWatchingForUpdates() {
    styleManagers.forEach((manager) => manager.pause());
    stopStylePositionWatchers();
    stopWatchingForStyleChanges();
    stopWatchingForInlineStyles();
    removeDOMReadyListener(onDOMReady);
    cleanReadyStateCompleteListeners();
}
let metaObserver;
function addMetaListener() {
    metaObserver = new MutationObserver(() => {
        if (document.querySelector('meta[name="darkreader-lock"]')) {
            metaObserver.disconnect();
            removeDynamicTheme();
        }
    });
    metaObserver.observe(document.head, {childList: true, subtree: true});
}
function createDarkReaderInstanceMarker() {
    const metaElement = document.createElement("meta");
    metaElement.name = "darkreader";
    metaElement.content = INSTANCE_ID;
    document.head.appendChild(metaElement);
}
function isAnotherDarkReaderInstanceActive() {
    if (document.querySelector('meta[name="darkreader-lock"]')) {
        return true;
    }
    const meta = document.querySelector('meta[name="darkreader"]');
    if (meta) {
        if (meta.content !== INSTANCE_ID) {
            return true;
        }
```

```
        return false;
    }
    createDarkReaderInstanceMarker();
    addMetaListener();
    return false;
}
function selectRelevantFix(documentURL, fixes) {
    if (!fixes) {
        return null;
    }
    if (fixes.length === 0 || fixes[0].url[0] !== "*") {
        return null;
    }
    const relevantFixIndex = findRelevantFix(documentURL, fixes);
    return relevantFixIndex
        ? combineFixes([fixes[0], fixes[relevantFixIndex]])
        : fixes[0];
}
function createOrUpdateDynamicTheme(
    filterConfig,
    dynamicThemeFixes,
    iframe
) {
    const dynamicThemeFix = selectRelevantFix(
        document.location.href,
        dynamicThemeFixes
    );
    createOrUpdateDynamicThemeInternal(
        filterConfig,
        dynamicThemeFix,
        iframe
    );
}
function createOrUpdateDynamicThemeInternal(
    filterConfig,
    dynamicThemeFixes,
    iframe
) {
    filter = filterConfig;
    fixes = dynamicThemeFixes;
    if (fixes) {
        ignoredImageAnalysisSelectors = Array.isArray(
            fixes.ignoreImageAnalysis
        )
            ? fixes.ignoreImageAnalysis
            : [];
        ignoredInlineSelectors = Array.isArray(fixes.ignoreInlineStyle)
            ? fixes.ignoreInlineStyle
            : [];
    } else {
        ignoredImageAnalysisSelectors = [];
        ignoredInlineSelectors = [];
    }
    if (filter.immediateModify) {
        setIsDOMReady(() => {
            return true;
        });
    }
    isIFrame = iframe;
    if (document.head) {
        if (isAnotherDarkReaderInstanceActive()) {
            removeDynamicTheme();
            return;
        }
        document.documentElement.setAttribute(
            "data-darkreader-mode",
            "dynamic"
        );
        document.documentElement.setAttribute(
```

```
            "data-darkreader-scheme",
            filter.mode ? "dark" : "dimmed"
        );
        createThemeAndWatchForUpdates();
    } else {
        {
            const fallbackStyle = createOrUpdateStyle(
                "darkreader--fallback"
            );
            document.documentElement.appendChild(fallbackStyle);
            fallbackStyle.textContent = getModifiedFallbackStyle(filter, {
                strict: true
            });
        }
        const headObserver = new MutationObserver(() => {
            if (document.head) {
                headObserver.disconnect();
                if (isAnotherDarkReaderInstanceActive()) {
                    removeDynamicTheme();
                    return;
                }
                createThemeAndWatchForUpdates();
            }
        });
        headObserver.observe(document, {childList: true, subtree: true});
    }
}
function removeProxy() {
    document.dispatchEvent(new CustomEvent("__darkreader__cleanUp"));
    removeNode(document.head.querySelector(".darkreader--proxy"));
}
const cleaners = [];
function removeDynamicTheme() {
    document.documentElement.removeAttribute(`data-darkreader-mode`);
    document.documentElement.removeAttribute(`data-darkreader-scheme`);
    cleanDynamicThemeCache();
    removeNode(document.querySelector(".darkreader--fallback"));
    if (document.head) {
        restoreMetaThemeColor();
        removeNode(document.head.querySelector(".darkreader--user-agent"));
        removeNode(document.head.querySelector(".darkreader--text"));
        removeNode(document.head.querySelector(".darkreader--invert"));
        removeNode(document.head.querySelector(".darkreader--inline"));
        removeNode(document.head.querySelector(".darkreader--override"));
        removeNode(document.head.querySelector(".darkreader--variables"));
        removeNode(document.head.querySelector(".darkreader--root-vars"));
        removeNode(document.head.querySelector('meta[name="darkreader"]'));
        removeProxy();
    }
    shadowRootsWithOverrides.forEach((root) => {
        removeNode(root.querySelector(".darkreader--inline"));
        removeNode(root.querySelector(".darkreader--override"));
    });
    shadowRootsWithOverrides.clear();
    forEach(styleManagers.keys(), (el) => removeManager(el));
    loadingStyles.clear();
    cleanLoadingLinks();
    forEach(document.querySelectorAll(".darkreader"), removeNode);
    adoptedStyleManagers.forEach((manager) => manager.destroy());
    adoptedStyleManagers.splice(0);
    adoptedStyleFallbacks.forEach((fallback) => fallback.destroy());
    adoptedStyleFallbacks.clear();
    stopWatchingPotentialAdoptedStyleNodes();
    metaObserver && metaObserver.disconnect();
    cleaners.forEach((clean) => clean());
    cleaners.splice(0);
}
function cleanDynamicThemeCache() {
    variablesStore.clear();
```

```
        parsedURLCache.clear();
        removeDocumentVisibilityListener();
        cancelRendering();
        stopWatchingForUpdates();
        cleanModificationCache();
        clearColorCache();
    }

    const blobRegex = /url\(\"(blob\:.*?)\"\)/g;
    async function replaceBlobs(text) {
        const promises = [];
        getMatches(blobRegex, text, 1).forEach((url) => {
            const promise = loadAsDataURL(url);
            promises.push(promise);
        });
        const data = await Promise.all(promises);
        return text.replace(blobRegex, () => `url("${data.shift()}")`);
    }
    const banner = `/*
                        _______
                       /       \\
                      .==.    .==.
                     ((  ))==((  ))
                    / "=="    "=="\\
                   /____|| || ||___\\
       ________     ____   ________  ___    ___
      |  ___  \\   /    \\  |  ___  \\ |  |  /  /
      |  |  \\  \\ /  /\\  \\  |  |  \\  \\|  |_/  /
      |  |   )  /  /__\\  \\ |  |__/  /|  ___  \\
      |  |__/  /  ______  \\|  ____  \\|  |  \\  \\
______|_______/__/ ____ \\__\\__|___\\__\\__|___\\__\\____
|  ___  \\ |  ____/ /    \\   |  ___  \\ |  ____|  ___  \\
|  |  \\  \\|  |___ /  /\\  \\  |  |  \\  \\|  |___|  |  \\  \\
|  |__/  /|  ____/  /__\\  \\ |  |   )  |  ____|  |__/  /
|  ____  \\|  |__/  ______  \\|  |__/  /|  |___|  ____  \\
|__|   \\__\\__\\____/__/      \\__\\_______/ |______|__|   \\__\\
                https://darkreader.org
*/

/*! Dark reader generated CSS | Licensed under MIT https://github.com/darkreader/darkreader/blob/main/LICENSE */
`;
    async function collectCSS() {
        const css = [banner];
        function addStaticCSS(selector, comment) {
            const staticStyle = document.querySelector(selector);
            if (staticStyle && staticStyle.textContent) {
                css.push(`/* ${comment} */`);
                css.push(staticStyle.textContent);
                css.push("");
            }
        }
        addStaticCSS(".darkreader--fallback", "Fallback Style");
        addStaticCSS(".darkreader--user-agent", "User-Agent Style");
        addStaticCSS(".darkreader--text", "Text Style");
        addStaticCSS(".darkreader--invert", "Invert Style");
        addStaticCSS(".darkreader--variables", "Variables Style");
        const modifiedCSS = [];
        document.querySelectorAll(".darkreader--sync").forEach((element) => {
            forEach(element.sheet.cssRules, (rule) => {
                rule && rule.cssText && modifiedCSS.push(rule.cssText);
            });
        });
        if (modifiedCSS.length) {
            const formattedCSS = formatCSS(modifiedCSS.join("\n"));
            css.push("/* Modified CSS */");
            css.push(await replaceBlobs(formattedCSS));
            css.push("");
        }
        addStaticCSS(".darkreader--override", "Override Style");
```

```
    return css.join("\n");
}

let unloaded = false;
const scriptId = generateUID();
function cleanup() {
    unloaded = true;
    removeEventListener("pagehide", onPageHide);
    removeEventListener("freeze", onFreeze);
    removeEventListener("resume", onResume);
    cleanDynamicThemeCache();
    stopDarkThemeDetector();
    stopColorSchemeChangeDetector();
}
function sendMessage(message) {
    if (unloaded) {
        return;
    }
    const responseHandler = (response) => {
        if (response === "unsupportedSender") {
            removeStyle();
            removeSVGFilter();
            removeDynamicTheme();
            cleanup();
        }
    };
    try {
        if (false);
        else {
            chrome.runtime.sendMessage(message, responseHandler);
        }
    } catch (error) {
        if (error.message === "Extension context invalidated.") {
            console.log(
                "Dark Reader: instance of old CS detected, clening up."
            );
            cleanup();
        } else {
            console.log(
                "Dark Reader: unexpected error during message passing."
            );
        }
    }
}
function onMessage(message) {
    if (
        message.scriptId !== scriptId &&
        message.type !== MessageTypeUItoCS.EXPORT_CSS
    ) {
        return;
    }
    logInfoCollapsed(`onMessage[${message.type}]`, message);
    switch (message.type) {
        case MessageTypeBGtoCS.ADD_CSS_FILTER:
        case MessageTypeBGtoCS.ADD_STATIC_THEME: {
            const {css, detectDarkTheme, detectorHints} = message.data;
            removeDynamicTheme();
            createOrUpdateStyle$1(
                css,
                message.type === MessageTypeBGtoCS.ADD_STATIC_THEME
                    ? "static"
                    : "filter"
            );
            if (detectDarkTheme) {
                runDarkThemeDetector((hasDarkTheme) => {
                    if (hasDarkTheme) {
                        removeStyle();
                        onDarkThemeDetected();
                    }
```

```
                }, detectorHints);
            }
            break;
        }
        case MessageTypeBGtoCS.ADD_SVG_FILTER: {
            const {
                css,
                svgMatrix,
                svgReverseMatrix,
                detectDarkTheme,
                detectorHints
            } = message.data;
            removeDynamicTheme();
            createOrUpdateSVGFilter(svgMatrix, svgReverseMatrix);
            createOrUpdateStyle$1(css, "filter");
            if (detectDarkTheme) {
                runDarkThemeDetector((hasDarkTheme) => {
                    if (hasDarkTheme) {
                        removeStyle();
                        removeSVGFilter();
                        onDarkThemeDetected();
                    }
                }, detectorHints);
            }
            break;
        }
        case MessageTypeBGtoCS.ADD_DYNAMIC_THEME: {
            const {theme, fixes, isIFrame, detectDarkTheme, detectorHints} =
                message.data;
            removeStyle();
            createOrUpdateDynamicTheme(theme, fixes, isIFrame);
            if (detectDarkTheme) {
                runDarkThemeDetector((hasDarkTheme) => {
                    if (hasDarkTheme) {
                        removeDynamicTheme();
                        onDarkThemeDetected();
                    }
                }, detectorHints);
            }
            break;
        }
        case MessageTypeUItoCS.EXPORT_CSS:
            collectCSS().then((collectedCSS) =>
                sendMessage({
                    type: MessageTypeCStoUI.EXPORT_CSS_RESPONSE,
                    data: collectedCSS
                })
            );
            break;
        case MessageTypeBGtoCS.UNSUPPORTED_SENDER:
        case MessageTypeBGtoCS.CLEAN_UP:
            removeStyle();
            removeSVGFilter();
            removeDynamicTheme();
            stopDarkThemeDetector();
            break;
    }
}
function sendConnectionOrResumeMessage(type) {
    sendMessage({
        type,
        scriptId,
        data: {
            isDark: isSystemDarkModeEnabled(),
            isTopFrame: window === window.top
        }
    });
}
runColorSchemeChangeDetector((isDark) =>
```

```
        sendMessage({
            type: MessageTypeCStoBG.COLOR_SCHEME_CHANGE,
            data: {isDark}
        })
    );
    chrome.runtime.onMessage.addListener(onMessage);
    sendConnectionOrResumeMessage(MessageTypeCStoBG.DOCUMENT_CONNECT);
    function onPageHide(e) {
        if (e.persisted === false) {
            sendMessage({type: MessageTypeCStoBG.DOCUMENT_FORGET, scriptId});
        }
    }
    function onFreeze() {
        sendMessage({type: MessageTypeCStoBG.DOCUMENT_FREEZE});
    }
    function onResume() {
        sendConnectionOrResumeMessage(MessageTypeCStoBG.DOCUMENT_RESUME);
    }
    function onDarkThemeDetected() {
        sendMessage({type: MessageTypeCStoBG.DARK_THEME_DETECTED});
    }
    {
        addEventListener("pagehide", onPageHide, {passive: true});
        addEventListener("freeze", onFreeze, {passive: true});
        addEventListener("resume", onResume, {passive: true});
    }
})();
```